QFESTA: Efficient Algorithms and Parameters for FESTA using Quaternion Algebras

Kohei Nakagawa[1] and Hiroshi Onuki[2]

[1] NTT Social Informatics Laboratories, Japan
[2] The University of Tokyo, Japan

Abstract. In 2023, Basso, Maino, and Pope proposed FESTA (Fast Encryption from Supersingular Torsion Attacks), an isogeny-based public-key encryption (PKE) protocol that uses the SIDH attack for decryption. In the same paper, they proposed a parameter for that protocol, but the parameter requires high-degree isogeny computations. In this paper, we introduce QFESTA (Quaternion Fast Encapsulation from Supersingular Torsion Attacks), a new variant of FESTA that works with better parameters using quaternion algebras and achieves IND-CCA security under QROM. To realize our protocol, we construct a new algorithm to compute an isogeny of non-smooth degree using quaternion algebra and the SIDH attack. Our protocol relies solely on $(2, 2)$-isogeny and 3-isogeny computations, promising a substantial reduction in computational costs. In addition, our protocol has significantly smaller data sizes for public keys and ciphertexts, approximately one-third the size of the original FESTA.

1 Introduction

In recent years, isogeny-based cryptography has been actively studied as one of the candidates for post-quantum cryptography (PQC). In particular, SIDH [24], proposed by De Feo, Jao, and Plut, is one of the well-known isogeny-based cryptosystems. Additionally, SIKE [2], a key encapsulation scheme based on SIDH, remained an alternative candidate for the NIST PQC standardization competition until Round 4. However, recent attacks [7,29,32] broke the security of SIDH and SIKE. These attacks find the secret isogeny from the two point images of the isogeny by computing $(2, 2)$-isogenies.

In response, a new isogeny-based cryptosystem called FESTA (Fast Encryption from Supersingular Torsion Attacks) [3], was proposed by Basso, Maino, and Pope, which is a public-key encryption (PKE) protocol that uses the SIDH attack for decryption. Their protocol seems resistant to the SIDH attack because it multiplies two point images by a random 2×2 matrix when generating the ciphertext. FESTA requires the computations of three isogenies: ϕ_A, ϕ_1, and ϕ_2 of degree d_A, d_1, and d_2, respectively. These three isogenies satisfy the following diagram.

$$\begin{tikzcd} E_0 \arrow[r, "\phi_A"] \arrow[d, "\phi_1"'] & E_A \arrow[d, "\phi_2"] \\ E_1 & E_2, \end{tikzcd}$$

Due to their construction, the degrees d_A, d_1, and d_2 must be *smooth* integers satisfying the following conditions:

$$d_A = d_{A,1}d_{A,2} \text{ and } m_1^2 d_{A,1}d_1 + m_2^2 d_{A,2}d_2 = 2^b$$

for some positive integers m_1, m_2, and b. These strong constraints make the parameters of FESTA much larger than SIDH, leading to larger data sizes of the public key and ciphertext. In addition, their protocol requires high-degree isogeny computations for key generation and encryption and is expected to be not efficient.

In this paper, we introduce a new PKE protocol based on FESTA that offers improved parameters and satisfies one-wayness against quantum plaintext checking oracle (OW-qPCA) security. The main innovation in our protocol is the use of our new algorithm named **RandIsogImages**, which computes the codomain and point images of a *non-smooth* degree d-isogeny from a special elliptic curve E_0. We construct this algorithm using a *quaternion technique* and the *SIDH attack*. We provide an overview of **RandIsogImages** below:

1. Let $\mathcal{O}_0 \cong \mathrm{End}(E_0)$, which is a maximal order of a quaternion algebras.
2. Let $D = 2^a$ such that $E_0[D] \subset E_0(\mathbb{F}_{p^2})$ and let P_0, Q_0 be a basis of $E_0[D]$.
3. Find $\alpha \in \mathcal{O}_0$ of norm $d \cdot (D - d)$.
4. Decompose $\alpha = \hat{\rho} \circ \tau$, where τ and ρ are isogenies of degree d and $D - d$, respectively.
5. Execute the SIDH attack by using the two point images $\alpha(P_0)$ and $\alpha(Q_0)$.
6. Obtain the codomain and the arbitrary point images of d-isogeny τ.

By using our new algorithm, the smoothness restriction for degrees d_A and d_1 are omitted, allowing for smaller parameters. Note that we must compute a $(2, 2)$-isogeny chain for this algorithm since we rely on the SIDH attack.

Additionally, to achieve indistinguishability against chosen ciphertext attack (IND-CCA) security under quantum random oracle model (QROM), we utilize the Fujisaki-Okamoto transform [20]. Consequently, our protocol functions as a key encapsulation mechanism (KEM) rather than PKE. We have named our new KEM 'QFESTA' (Quaternion Fast Encapsulation from Supersingular Torsion Attacks).

As mentioned above, the removal of the smoothness restriction allows us to use more efficient parameters. In fact, our protocol uses less than a quarter of the characteristic p and approximately one third of the public key and ciphertext size compared to the original FESTA under the NIST security level 1, 3, and 5. Moreover, our protocol only requires $(2, 2)$-isogeny and 3-isogeny computations and is expected to be highly efficient, whereas the original FESTA requires high-degree isogeny computations. (See Table 1.)

	FESTA	**QFESTA**
KeyGen	isogenies of degrees 59 to 41161	(2,2)-isogenies
Enc	isogenies of degrees 3 to 3779	3-isogneies and (2,2)-isogenies
Dec	(2,2)-isogenies	(2,2)-isogenies

Table 1: Isogeny computations in FESTA and QFESTA for the NIST security level 1.

Remark 1. The recent paper [9] proposed a polynomial-time attack on FESTA. However, their attack succeeds only when the basis P_0, Q_0 of $E_0[2^b]$, which is a system parameter of FESTA, satisfies a specific condition. According to their paper, the probability of randomly chosen $P_0, Q_0 \in E_0[2^b]$ satisfying the condition is sufficiently small. Moreover, we can determine in polynomial time whether a basis is “safe”, meaning that the computational cost of their attack against FESTA with the basis is in $O(2^\lambda)$. Our implementation of QFESTA includes an algorithm to make that decision and can be found at: `https://github.com/hiroshi-onuki/QFESTA-SageMath`.

1.1 Contributions

In this paper, we make the following contributions:

1. We construct the new algorithm **RandIsogImages**, which computes the codomain and point images of a non-smooth degree isogeny from a special elliptic curve E_0.
2. Using our new algorithm **RandIsogImages**, we propose a new PKE that has less data sizes and less computational cost than FESTA.
3. We prove that our PKE is OW-qPCA secure. The security proof relies on novel security assumptions that are variants of FESTA's security assumptions.
4. By applying the Fujisaki-Okamoto transform to our PKE, we obtain a new KEM that is IND-CCA secure under QROM. We call this KEM ‘QFESTA’.
5. We describe a method to find parameters for QFESTA and give concrete parameters for the NIST security level 1, 3, and 5. Under these parameter settings, we analyse the public key and ciphertext sizes.
6. Finally, we implement the proposed QFESTA in SageMath [34] as a proof-of-concept and compare the computational cost with FESTA.

1.2 Organization

In Section 2, we give some notation and background knowledge used in our protocol. In Section 3, we propose our new KEM named QFESTA and its security is analysed in Section 4. In Section 5, we give some concrete parameters for QFESTA and analyse the data size and the computational cost of QFESTA under a proof-of-concept implementation. Finally, in Section 6, we give the conclusion of this paper.

2 Preliminaries

In this section, we summarise some background knowledge used in our protocol.

2.1 Notation

Throughout this paper, we use the following notation. We let p be a prime number of cryptographic size, i.e., p is at least about 2^{256}. Let $f(x)$ and $g(x)$ be real functions. We write $f(x) = O(g(x))$ if there exists a constant $c \in \mathbb{R}$ such that $f(x)$ is bounded by $c \cdot g(x)$ for sufficiently large x. $f(x)$ is *negligible* if $|f(x)| < x^{-c}$ for all positive integers c and sufficiently large x. We write $f(x) < \mathrm{negl}(x)$ if $f(x)$ is negligible. For a finite set S, we write $x \in_U S$ if x is sampled uniformly at random from S. Let $\perp$ be the symbol indicating failure of an algorithm.

2.2 Isogenies

In this paper, we mainly use principally polarized superspecial abelian varieties defined over a finite field of characteristic p of dimension one or two. Such a variety is isomorphic to a supersingular elliptic curve, the product of two supersingular elliptic curves, or a jacobian of a superspecial hyperelliptic curve of genus two, and always has a model defined over $\mathbb{F}_{p^2}$. Therefore, we only consider varieties defined over $\mathbb{F}_{p^2}$.

Basic Facts. An *isogeny* is a rational map between abelian varieties which is a surjective group homomorphism and has finite kernel. The *degree* of an isogeny φ is its degree as a rational map and denoted by $\deg \varphi$. An isogeny φ is *separable* if $\# \ker \varphi = \deg \varphi$. A separable isogeny is uniquely determined by its kernel up to post-composition of isomorphism. For an isogeny $\varphi : A \to B$ between principally polarized abelian varieties, there exists a unique *dual isogeny* $\hat{\varphi}$ such that $\hat{\varphi} \circ \varphi$ is equal to the multiplication-by-$\deg \varphi$ map on A.

Let A and B be principally polarized abelian varieties. If there exists an isogeny between A and B then the dimensions of A and B are the same. If A is superspecial then there exists an isogeny between A and B if and only if B is a superspecial abelian variety of the same dimension as A.

Let A be a principally polarized abelian variety and ℓ a positive integer. An *ℓ-isotropic subgroup* of A is a subgroup of the ℓ-torsion subgroup $A[\ell]$ of A on which the ℓ-Weil pairing is trivial. An ℓ-isotropic subgroup G is *maximal* if there is no other ℓ-isotropic subgroup containing G. A separable isogeny whose kernel is a maximal ℓ-isotropic subgroup is called an *ℓ-isogeny* if the dimension of the domain is one or an *(ℓ, ℓ)-isogeny* if the dimension of the domain is two.

Computing Isogenies. Let A be a principally polarized abelian variety, ℓ a positive integer, and G a maximal ℓ-isotropic subgroup of A.

If the dimension of A is one then we can compute an ℓ-isogeny φ with kernel G by Vélu's formulas [36]. More precisely, given A, ℓ, G, Vélu's formulas give a method to compute the codomain of φ in $O(\ell)$ operations on a field containing the points in G. In addition, for additional input $P \in A$, we can compute $\varphi(P)$ in $O(\ell)$ operations on a field containing the points in G and P. These computational costs are improved to $\tilde{O}(\sqrt{\ell})$ by Bernstein, De Feo, Leroux, and Smith [5].

If A is the jacobian of a hyperelliptic curve of genus two and $\ell = 2$ then we can compute $(2,2)$-isogeny by formulas in Smith's Ph.D thesis [33]. Formulas of $(2,2)$-isogenies for the case A is the product of two elliptic curves is given by Howe, Leprévost, and Poonen [23]. Cosset and Robert [11] gave a method to compute (ℓ,ℓ)-isogenies for general ℓ. The computational cost of their method is $O(\ell^4)$ operations on a field containing the points in G.

2.3 Quaternion Algebras and the Deuring Correspondence

Quaternion Algebras. A *quaternion algebra* over $\mathbb{Q}$ is a division algebra defined by $\mathbb{Q} + \mathbb{Q}\mathbf{i} + \mathbb{Q}\mathbf{j} + \mathbb{Q}\mathbf{k}$ and $\mathbf{i}^2 = a, \mathbf{j}^2 = b, \mathbf{ij} = -\mathbf{ji} = \mathbf{k}$ for $a, b \in \mathbb{Q}^*$. We denote it by $H(a,b)$. We say $H(a,b)$ is *ramified* at a place v of $\mathbb{Q}$ if $H(a,b) \otimes_{\mathbb{Q}} \mathbb{Q}_v$ is not isomorphic to the algebra of the 2×2 matrices over $\mathbb{Q}_v$. There exists a quaternion algebra ramified exactly at p and ∞. Such an algebra is unique up to isomorphism. We denote it by $\mathcal{B}_{p,\infty}$.

Let $\alpha = x + y\mathbf{i} + z\mathbf{j} + t\mathbf{k} \in H(a,b)$ with $x, y, z, t \in \mathbb{Q}$. The *canonical involution* of α is $x - y\mathbf{i} - z\mathbf{j} - t\mathbf{k}$ and denoted by $\bar{\alpha}$. The *reduced norm* of α is $\alpha\bar{\alpha}$ and denoted by $n(\alpha)$.

An *order* $\mathcal{O}$ of $H(a,b)$ is a subring of $H(a,b)$ that is also a $\mathbb{Z}$-lattice of rank 4. This means that $\mathcal{O} = \mathbb{Z}\alpha_1 + \mathbb{Z}\alpha_2 + \mathbb{Z}\alpha_3 + \mathbb{Z}\alpha_4$ for a basis $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ of $H(a,b)$. We denote such an order by $\mathbb{Z}\langle\alpha_1, \alpha_2, \alpha_3, \alpha_4\rangle$. An order $\mathcal{O}$ is said to be *maximal* if there is no larger order that contains $\mathcal{O}$.

Deuring Correspondence. Deuring [16] showed that the endomorphism ring of a supersingular elliptic curve over $\mathbb{F}_{p^2}$ is isomorphic to a maximal order of $\mathcal{B}_{p,\infty}$ and gave a correspondence (*Deuring correspondence*) where a supersingular elliptic E curve over $\mathbb{F}_{p^2}$ corresponds to a maximal order isomorphic to $\mathrm{End}(E)$.

Suppose $p \equiv 3 \pmod 4$. This is the setting we use in our protocol. Then we can take $\mathcal{B}_{p,\infty} = H(-1,-p)$ and an elliptic curve over $\mathbb{F}_{p^2}$ with j-invariant 1728 is supersingular. Let E_0 be the elliptic curve over $\mathbb{F}_{p^2}$ defined by $y^2 = x^3 + x$. Then $j(E_0) = 1782$ so E_0 is supersingular. We define endomorphisms $\iota : (x,y) \mapsto (-x, \sqrt{-1}y)$ and $\pi : (x,y) \mapsto (x^p, y^p)$ of E_0, where $\sqrt{-1}$ is a fixed square root of -1 in $\mathbb{F}_{p^2}$. The endomorphism ring of E_0 is isomorphic to $\mathcal{O}_0 \coloneqq \mathbb{Z}\langle 1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2}\rangle$. This isomorphism is given by $\iota \mapsto \mathbf{i}$ and $\pi \mapsto \mathbf{j}$. From now on, we identify $\mathrm{End}(E_0)$ with $\mathcal{O}_0$ by this isomorphism.

Some isogeny-based protocols, e.g., SQISign [13], need to compute the image under an element in $\mathcal{O}_0$ represented by the coefficients with respect to the basis $(1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2})$. Let $P \in E_0(\mathbb{F}_{p^2})$ and $\alpha = x + y\mathbf{i} + z\frac{\mathbf{i}+\mathbf{j}}{2} + t\frac{1+\mathbf{k}}{2}$ for $x, y, z, t \in \mathbb{Z}$. Given P and x, y, z, t, one can compute $\alpha(P)$ in $O(\log \max\{|x|, |y|, |z|, |t|\})$ operations on $\mathbb{F}_{p^2}$ and $O(\log p)$ operations on $\mathbb{F}_{p^4}$. The latter operations on $\mathbb{F}_{p^4}$ is necessary only for the case the order of P is even. We need to compute $\alpha(P_0)$

and $\alpha(Q_0)$ for a fixed basis P_0, Q_0 of $E_0[2^a]$ for some a in our protocol. In this case, by precomputing the images of P_0 and Q_0 under $\mathbf{i}$, $\frac{\mathbf{i}+\mathbf{j}}{2}$, and $\frac{1+\mathbf{k}}{2}$, we can compute $\alpha(P_0)$ and $\alpha(Q_0)$ by scalar multiplications by x, y, z, t and additions.

Computing Quaternion with Given Norm. As in the above, we let $\mathcal{O}_0$ be the maximal order of $\mathcal{B}_{p,\infty}$ with basis $(1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2})$. We need an algorithm to compute an element in $\mathcal{O}_0$ of given norm in our protocol. We can use an algorithm **RepresentInteger** proposed by Kohel, Lauter, Petit, and Tignol [27]. **RepresentInteger** takes an integer $M > p$ as input and outputs $\alpha \in \mathbb{Z}\langle 1, \mathbf{i}, \mathbf{j}, \mathbf{k}\rangle \subset \mathcal{O}_0$ such that $n(\alpha) = M$. Later, De Feo, Leroux, Longa, and Wesolowski [14] extended **RepresentInteger** to take output from all elements in $\mathcal{O}_0$. They named the new algorithm **FullRepresentInteger**.

Algorithm 2 gives a pseudocode of **FullRepresentInteger**. This uses Cornacchia's algorithm [12, Algorithm 2.3.12], which takes a prime q as input and outputs integers x, y such that $x^2 + y^2 = q$ or $\perp$ if such integers do not exist. One can extends this to take a positive integer as input by using a well-known relation: $(x^2 + y^2)(z^2 + t^2) = (xz - yt)^2 + (xt + yz)^2$. This extension requires the prime factorization of the input. In general, the computational time of the prime factorization is subexponential in the size of the input. To make our algorithm work in polynomial time in $\log p$, we use "pseudo-factorization" in our algorithm. In particular, our extension of Cornacchia's algorithm with input M returns x, y such that $x^2 + y^2 = M$ if and only if such integers exists and M is the product of a smooth number and a prime. This method is used in SQIsign (see the official document [10] for the detail). We denote this alternate version of Cornacchia's algorithm by **Cornachhia**. Due to the failure of the factorization, the outputs of Algorithm 1 does not contain all elements in $\mathcal{O}_0$ whose norm is the input M. However, from the prime number theorem, we can assume at least $1/\log M$ of all elements in $\mathcal{O}_0$ whose norm is the input M could be the output of Algorithm 1.

2.4 Computing Isogenies of Dimension one from Dimension Two

In this subsection, we give algorithms to compute isogenies of dimension one by using an isogeny of dimension two, which are a main sub-algorithm for FESTA and our protocol. These algorithms are comes from recent attacks to SIDH by [7,28,32]. We use the following theorem, which is based on Kani's criterion [26].

Theorem 1 ([28, Theorem 1]). *Let N_1, N_2, and D be pairwise coprime integers such that $D = N_1 + N_2$, and let E_0, E_1, E_2, and E_3 be elliptic curves connected by the following commutative diagram of isogenies:*

$$\begin{tikzcd} E_0 \arrow[r, "\psi_2"] \arrow[d, "\psi_1"'] & E_2 \arrow[d, "\psi_1'"] \\ E_1 \arrow[r, "\psi_2'"'] \arrow[ur, "f"] & E_3, \end{tikzcd}$$

Algorithm 1 FullRepresentInteger$_{\mathcal{O}_0}(M)$

```
Require: An integer $M > p$.
Ensure: $\alpha \in \mathcal{O}_0$ such that $n(\alpha) = M$.
 1: Let $m' = \lfloor\sqrt{\frac{4M}{p}}\rfloor$ and sample a random integer $z' \in [-m', m']$.
 2: Let $m'' = \lfloor\sqrt{\frac{4M}{p} - z'^2}\rfloor$ and sample a random integer $t' \in [-m'', m'']$.
 3: Let $M' = 4M - p(z'^2 + t'^2)$.
 4: if Cornachhia$(M') = \perp$ then
 5:     Go back to Step 1.
 6: else
 7:     Set $(x', y') \leftarrow$ Cornachhia$(M')$.
 8: end if
 9: if $x' \not\equiv t' \mod 2$ or $y' \not\equiv z' \mod 2$ then
10:     Go back to Step 1.
11: end if
12: return $(x' + y'\mathbf{i} + z'\mathbf{j} + t'\mathbf{k})/2$.
```

where $\deg(\psi_1) = \deg(\psi'_1) = N_1$ *and* $\deg(\psi_2) = \deg(\psi'_2) = N_2$*. Then, the isogeny*

$$\Phi = \begin{pmatrix} \hat{\psi}_1 & -\hat{\psi}_2 \\ \psi'_2 & \psi'_1 \end{pmatrix} : E_1 \times E_2 \to E_0 \times E_3 \tag{1}$$

is a (D, D)*-isogeny with respect to the natural product polarizations on* $E_1 \times E_2$ *and* $E_0 \times E_3$*, and has kernel* $\{([N_2]P, f(P)) \mid P \in E_1[D]\}$*.*

Conversely, a (D, D)-isogeny with kernel $\{([N_2]P, f(P)) \mid P \in E_1[D]\}$ is of the form $\iota \circ \Phi$ with an isomorphism ι from $E_0 \times E_3$. To construct algorithms to evaluate the isogenies in the matrix in Equation (1), we need to restrict the possibility of ι. In particular, we assume that the codomain E_3 of ψ'_1 and ψ'_2 is not isomorphism to E_0. This assumption is plausible because there exist about $p/12$ supersingular elliptic curves over $\mathbb{F}_{p^2}$ up to isomorphism and ψ'_1 seems to be a random isogeny unless we intend to $E_1 \cong E_3$. Under this assumption, an isomorphism from $E_0 \times E_3$ is represented by $\begin{pmatrix} \iota_0 & 0 \\ 0 & \iota_3 \end{pmatrix}$ or $\begin{pmatrix} 0 & \iota_3 \\ \iota_0 & 0 \end{pmatrix}$, where ι_0 is an isomorphism from E_0 and ι_3 is an isomorphism from E_3.

Using Theorem 1 and assuming the above assumption, we construct two algorithms to evaluate the isogenies in the matrix in Equation (1) by computing a (D, D)-isogeny.

The first algorithm is for the case that we know E_0 in advance and denoted by **EvalByKani**. Let N_1, N_2 be integers coprime with each other and $D = N_1 + N_2$. Let E_0, E_1, E_2 supersingular elliptic curves over $\mathbb{F}_{p^2}$, (P_1, Q_1) a basis of $E_1[D]$, (P_2, Q_2) a basis of $E_2[D]$, S_1 a finite subset of E_1, and S_2 a finite subset of E_2. If there exist isogenies $\psi_1 : E_0 \to E_1$ and $\psi_2 : E_0 \to E_2$ such that $\deg \psi_1 = N_1$ $\deg \psi_2 = N_2$, $P_2 = \psi_2 \circ \hat{\psi}_1(P_1)$, and $Q_2 = \psi_2 \circ \hat{\psi}_1(Q_1)$, then **EvalByKani** with input $(N_1, N_2, E_0, E_1, E_2, P_1, Q_1, P_2, Q_2, S_1, S_2)$ returns the image of S_1 under $\hat{\psi}_1$ and the image of S_2 under $\hat{\psi}_2$. If such isogenies do not exist then **EvalByKani** returns $\perp$. The procedure for **EvalByKani** is as follows:

1. Compute a (D, D)-isogeny Φ with kernel $\langle([N_2]P_1, P_2), ([N_2]Q_1, Q_2)\rangle$.
2. If the codomain of Φ is not the product of elliptic curves then return $\perp$.
3. Otherwise let $F_1 \times F_2$ be the codomain of Φ.
4. If both of F_1 and F_2 are not isomorphic to E_0 then return $\perp$.
5. Otherwise change Φ so that the first component of the codomain is E_1 by composing an isomorphism.
6. Return the first components of $\Phi((R_1, O_{E_2}))$ and $\Phi((O_{E_1}, R_2))$ for $R_1 \in S_1$ and $R_2 \in S_2$, where O_E is the neutral element of E for an elliptic curve E.

We use the same notation as in the previous paragraph. The second algorithm **CodomainByKani** is for the case that we do not know the codomain of Φ and that there exists an integer $M > \max\{N_1, N_2\}$ coprime with N_1 and N_2 such that S_1 contains a basis of $E_1[M]$ or S_2 contains a basis of $E_2[M]$.

In this case, we can determine the order of elliptic curves in the codomain of Φ by computing the M-Weil pairing. More precisely, we use the following fact. For a basis R, T of $E_1[M]$ and an isogeny $\phi : E_1 \to F$, the M-Weil pairings $e_M(R, T)$ and $e_M(\phi(R), \phi(T))$ satisfying $e_M(R, T)^{\deg \phi} = e_M(\phi(R), \phi(T))$. This determines $\deg \phi \bmod M$.

The input of **CodomainByKani** is that of **EvalByKani** minus E_0 and the output of **CodomainByKani** is that of **EvalByKani** plus E_0. The procedure for **CodomainByKani** is the same in **EvalByKani** until Step 3. We describe the rest of the procedure in the case that $S_1 = E_1[D]$ and $S_2 = \emptyset$ for simplicity (this is the case we need in our protocol).

4. Let R, T be a basis of $E_1[D]$ in S_1.
5. Let $(R'_1, R'_2) = \Phi((R, O_{E_2}))$ and $(T'_1, T'_2) = \Phi((T, O_{E_2}))$.
6. Compute the D-Weil pairings $e_D(R, T)$ and $e_D(R'_1, T'_1)$.
7. If $e_D(R, T)^{N_1} = e_D(R'_1, T'_1)$ then return F_1 and (R'_1, T'_1). Otherwise return F_2 and (R'_2, T'_2).

When D is smooth, $P_1, Q_1 \in E_1(\mathbb{F}_{p^2})$, $S_1 \subset E_1(\mathbb{F}_{p^2})$, $P_2, Q_2 \in E_2(\mathbb{F}_{p^2})$, and $S_2 \subset E_2(\mathbb{F}_{p^2})$ the computational costs of **EvalByKani** and **CodomainByKani** are $O((\#S_1 + \#S_2) \log D)$ operations on $\mathbb{F}_{p^2}$ by using the methods stated in Section 2.2. Especially, D is a power of 2 in our case.

2.5 Cryptographic Preliminaries

In this subsection, we recall cryptographic notation, which is necessary for describing our protocol.

First, we define two cryptographic schemes, public key encryption (PKE) and key encapsulation mechanism (KEM).

Definition 1 (Public Key Encryption (PKE)). *A* public key encryption *consists of a set of parameters* $\{\mathbf{param}_\lambda\}_{\lambda \in \mathbb{Z}_{>0}}$*, a family of finite sets* $\{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{Z}_{>0}}$*, and three polynomial-time algorithms* **KeyGen**, **Enc***, and* **Dec** *such that*

– **KeyGen** *takes* $\mathbf{param}_\lambda$ *as input and outputs a pair* (pk, sk) *of keys,*

– **Enc** *takes* $\mathbf{param}_\lambda$*, a public key* pk*, and a message* $m \in \mathcal{M}_\lambda$ *as input and outputs a ciphertext* ct*,*
– *and* **Dec** *takes* $\mathbf{param}_\lambda$*, a secret key* sk*, and* ct *as input and outputs the message* m *if* ct *is a valid ciphertext or* $\perp$ *otherwise.*

Definition 2 (Key Encapsulation Mechanism (KEM)). *A* key encapsulation mechanism *consists of a set of parameters* $\{\mathbf{param}_\lambda\}_{\lambda\in\mathbb{Z}_{>0}}$ *and three polynomial-time algorithms* **KeyGen***,* **Encaps***, and* **Decaps** *such that*

– **KeyGen** *takes* $\mathbf{param}_\lambda$ *as input and outputs a pair* (pk, sk) *of keys,*
– **Encaps** *takes a public key* pk *as input and outputs a pair* (K, ct) *of a key and a ciphertext,*
– *and* **Decaps** *takes* $\mathbf{param}_\lambda$*, a secret key* sk*, and* ct *as input and outputs the key* K *if* ct *is a valid ciphertext or* $\perp$ *otherwise.*

For simplifying the notation, we omit $\mathbf{param}_\lambda$ from input of each algorithm. In particular, we denote $\mathbf{KeyGen}(\lambda), \mathbf{Enc}(pk, m)$, and so on.

Next, we define security notation, One-Wayness against quantum Plaintext Checking Attacks (OW-qPCA) for PKE and INDistinguishability against Chosen Ciphertext Attacks (IND-CCA) for KEM.

Definition 3 (OW-qPCA). *Let* $\Pi = (\{\mathbf{param}_\lambda\}, \{\mathcal{M}_\lambda\}, \mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ *be a PKE. Let* **Pco** *be an oracle such that* $\mathbf{Pco}(m', ct')$ *returns whether* $\mathbf{Dec}(sk, ct') = m'$ *for any* ct' *and* $m' \in \mathcal{M}_\lambda$*. We say that* Π *is* OW-qPCA *secure if, for any probabilistic polynomial-time adversary* $\mathcal{A}^{\mathrm{q}\mathbf{Pco}(\cdot,\cdot)}$ *who can make quantum queries to* **Pco***,*

$$\Pr\left[m = m^* \;\middle|\; \begin{array}{l} (pk, sk) \leftarrow \mathbf{KeyGen}(\lambda),\ m \in_U \mathcal{M}_\lambda, \\ ct \leftarrow \mathbf{Enc}(pk, m),\ m^* \leftarrow \mathcal{A}^{\mathrm{q}\mathbf{Pco}(\cdot,\cdot)}(pk, ct) \end{array}\right] < \mathrm{negl}(\lambda).$$

Definition 4 (IND-CCA). *Let* $\Pi = (\{\mathbf{param}_\lambda\}, \mathbf{KeyGen}, \mathbf{Encaps}, \mathbf{Decaps})$ *be a KEM and* $\mathcal{K}_\lambda$ *the set of the keys which* **Encaps** *with* $\mathbf{param}_\lambda$ *outputs. Let* **Cco** *be an oracle such that* $\mathbf{Cco}(ct')$ *returns* $\mathbf{Decaps}(sk, ct')$ *for any* $ct' \neq ct$*. We say that* Π *is* IND-CCA *secure if, for any probabilistic polynomial-time adversary* $\mathcal{A}^{\mathbf{Cco}(\cdot)}$ *who can make queries to* **Cco***,*

$$\left|\Pr\left[b = b^* \;\middle|\; \begin{array}{l} (pk, sk) \leftarrow \mathbf{KeyGen}(\lambda),\ b \in_U \{0, 1\}, \\ (K_0, ct) \leftarrow \mathbf{Encaps}(pk),\ K_1 \in_U \mathcal{K}_\lambda, \\ b^* \leftarrow \mathcal{A}^{\mathbf{Cco}(\cdot)}(pk, ct, K_b) \end{array}\right] - \frac{1}{2}\right| < \mathrm{negl}(\lambda).$$

Cryptographic Transform. The Fujisaki-Okamoto transforms [20] are methods to transform a cryptographic protocol with “weak” security into that with “strong” security by using cryptographic hash functions. In this paper, we use $\mathbf{U}^{\not\perp}$ transform in [25], which transforms an OW-qPCA PKE into an IND-CCA KEM under the quantum random oracle model (QROM).

Let $\Pi = (\{\mathbf{param}_\lambda\}, \{\mathcal{M}_\lambda\}, \mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ be a PKE, and $H = \{H_\lambda\}$ be a set of cryptographic hash functions such that $H_\lambda : \{0, 1\}^* \to \mathcal{M}_\lambda$. Then we define a KEM $\Pi^{\mathrm{FO}} := \mathsf{U}^{\not\perp}(\Pi, H)$ as follows. The parameter sets of Π^{FO} are the same as Π. **KeyGen** of Π^{FO} outputs a dummy message $s \in_U \mathcal{M}_\lambda$ in addition to the output of **KeyGen** of Π. A secret key of Π^{FO} is a pair (sk, s). **Encaps** and **Decaps** of Π^{FO} are defined as follows:

- **Encaps**$(pk) \to (K, (ct, d))$:
  1. $m \in_U \mathcal{M}_\lambda$.
  2. $ct \leftarrow \mathbf{Enc}(pk, m)$.
  3. $K = H_\lambda(m, ct)$.
  4. Return K, ct.
- **Decaps**$((sk, s), ct) \to K$:
  1. $m' \leftarrow \mathbf{Dec}(sk, ct)$.
  2. If $m' = \perp$ then return $H_\lambda(s, ct)$.
  3. Else return $H_\lambda(m', ct)$.

In the above setting, we can obtain an IND-CCA KEM from an IND-qPCA PKE.

Theorem 2 (Theorem 4 in [25]). *We use the above notation. If Π is OW-qPCA secure then Π^{FO} is IND-CCA secure against a quantum adversary under the assumption that the hash functions in H are quantum random oracles.*

2.6 FESTA

FESTA is an isogeny-based protocol proposed by Basso, Maino, and Pope [3]. This protocol is a PKE that uses **EvalByKani** for decryption. More precisely, Basso et al. constructed a trapdoor one-way function (*FESTA trapdoor function*) and obtained IND-CCA secure PKE by applying Optimal Asymmetric Encryption Padding (OAEP) transform [4]. In this subsection, we give an overview of FESTA. For the detail, see [3].

FESTA Trapdoor Function. The core idea of FESTA is as follows. Let E_0, E_A, E_1, E_2 be supersingular elliptic curves over $\mathbb{F}_{p^2}$ and P_0, Q_0 a basis of $E_0[n]$ for a positive integer n. Let $\phi_A : E_0 \to E_A$, $\phi_1 : E_0 \to E_1$, and $\phi_2 : E_A \to E_2$ be isogenies of degrees coprime with and less than n. If one knows the images of P_0 and Q_0 under one of the isogenies then the SIDH attacks in §2.4 reveals the isogeny. To prevent this attack, images "masked" by matrices are published in FESTA. More precisely, for 2×2 regular matrices $\mathbf{A}$ and $\mathbf{B}$ over $\mathbb{Z}/n\mathbb{Z}$, points $(R_A, S_A) := \mathbf{A}(\phi_A(P_0), \phi_A(Q_0))^\top$, $(R_1, S_1) := \mathbf{B}(\phi_1(P_0), \phi_1(Q_0))^\top$, and $(R_2, S_2) := \mathbf{B}(\phi_2(R_A), \phi_2(S_A))^\top$ are published (see Figure 1). Since the action of a matrix commutes with an isogeny, easy computation shows that

$$\begin{pmatrix} R_2 \\ S_2 \end{pmatrix} = \frac{1}{\deg \phi_1} \mathbf{B}\mathbf{A}\mathbf{B}^{-1} \phi_2 \circ \phi_A \circ \hat{\phi}_1 \begin{pmatrix} R_1 \\ S_1 \end{pmatrix}.$$

If $\mathbf{AB} = \mathbf{BA}$ then one can remove $\mathbf{B}$ from the right-hand side of the above equation. This means that the images of R_1 and S_1 under $\phi_2 \circ \phi_A \circ \hat{\phi}_1$ can be

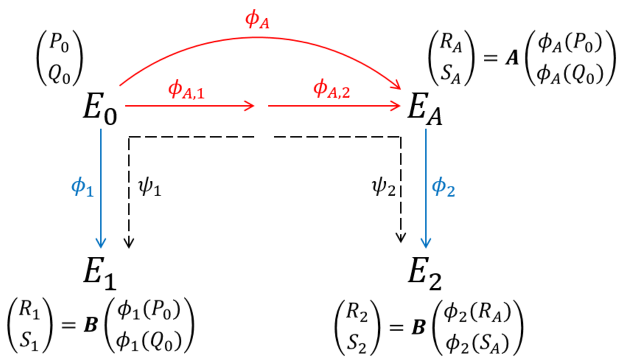
$\begin{pmatrix} P_0 \\ Q_0 \end{pmatrix}$
ϕ_A
$\begin{pmatrix} R_A \\ S_A \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} \phi_A(P_0) \\ \phi_A(Q_0) \end{pmatrix}$
E_0
$\phi_{A,1}$
$\phi_{A,2}$
E_A
ϕ_1
ψ_1
ψ_2
ϕ_2
E_1
E_2
$\begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = \boldsymbol{B} \begin{pmatrix} \phi_1(P_0) \\ \phi_1(Q_0) \end{pmatrix}$
$\begin{pmatrix} R_2 \\ S_2 \end{pmatrix} = \boldsymbol{B} \begin{pmatrix} \phi_2(R_A) \\ \phi_2(S_A) \end{pmatrix}$

Fig. 1: A picture of FESTA.

computed by using only $\mathbf{A}$. FESTA trapdoor function is a trapdoor function with secret key $\mathbf{A}$. In particular, the matrices $\mathbf{A}$ and $\mathbf{B}$ are chosen from diagonal matrices in the implementation by [3]. Therefore, we only consider this case in this paper.

To define FESTA trapdoor function precisely, we define system parameters for it. Let d_A, d_1, d_2 be smooth positive odd integers coprime with each other and lager than $2^{2\lambda}$ for a security parameter λ. These are the degrees of ϕ_A, ϕ_1, and ϕ_2, respectively. Let $d_{A,1}, d_{A,2}, m_1, m_2, b$ be positive integers satisfying the following condition:

$$d_A = d_{A,1}d_{A,2} \text{ and } m_1^2 d_{A,1}d_1 + m_2^2 d_{A,2}d_2 = 2^b.$$

In this setting, we let $p = d_1 d_2 (d_A)_{\mathrm{sf}} f - 1$ for a small cofactor f, where $(d_A)_{\mathrm{sf}}$ is the square-free part of d_A. This choice of p enable us to compute the isogenies in FESTA by operations over $\mathbb{F}_{p^2}$. We write $\mathcal{M}_n$ to denote the set of 2×2 diagonal regular matrices over $\mathbb{Z}/n\mathbb{Z}$, and let

$$\mathcal{E}_{p,2^b} = \left\{ (E, (P,Q)) \;\middle|\; \begin{array}{l} E: \text{ a supersingular elliptic curve over } \mathbb{F}_{p^2} \\ \text{such that } \#E(\mathbb{F}_{p^2}) = (p+1)^2, \\ (P,Q): \text{ a basis of } E[2^b] \end{array} \right\}.$$

As in the first paragraph of this subsubsection, let $(E_0, (P_0, Q_0)) \in \mathcal{E}_{p,2^b}$, $\phi_A : E_0 \to E_A$ be an isogeny of degree d_A, and $(R_A, S_A)^\top = \mathbf{A}\phi_A(P_0, Q_0)^\top$. In addition, we choose and publish bases (K_0, K_0') and (K_A, K_A') of $E_0[d_1]$ and $E_A[d_2]$ for computing generators of the kernels of secret isogenies. The FESTA trapdoor function with public information E_A, R_A, S_A is a function

$$f_{(E_A,R_A,S_A)} : \mathbb{Z}/d_1\mathbb{Z} \times \mathbb{Z}/d_2\mathbb{Z} \times \mathcal{M}_{2^b} \to \mathcal{E}_{p,2^b} \times \mathcal{E}_{p,2^b}.$$

The secret key of this function is $\mathbf{A}$ and ϕ_A. The output of $f_{(E_A,R_A,S_A)}$ with input $(n_1, n_2, \mathbf{B})$ is computed as follows.

1. Compute an isogeny $\phi_1 : E_0 \to E_1$ with kernel $\langle K_0 + [n_1]K_0' \rangle$.
2. Compute an isogeny $\phi_2 : E_A \to E_2$ with kernel $\langle K_A + [n_2]K_A' \rangle$.
3. Ouput $(E_1, \mathbf{B}(\phi_1(P_0), \phi_2(Q_0))^\top)$ and $(E_2, \mathbf{B}(\phi_2(R_A), \phi_2(S_A))^\top)$.

Anyone who knows the secret key $(\mathbf{A}, \phi_A)$ can compute the inverse of this function by using **EvalByKani**. We decompose ϕ_A into $\phi_{A,1}$ and $\phi_{A,2}$ of degrees $d_{A,1}$ and $d_{A,2}$, respectively. Let F be the codomain of $\phi_{A,1}$. We define $\psi_1 := [m_1] \circ \phi_1 \circ \hat{\phi}_{A,1}$, $\psi_2 := [m_2] \circ \phi_1 \circ \phi_{A,2}$, and $f := \psi_2 \circ \hat{\psi}_1 = [m_1 m_2] \circ \phi_2 \circ \phi_A \circ \hat{\phi}_1$ (see Figure 1). Then we have $\deg \psi_1 + \deg \psi_2 = 2^b$ and $[m_1 m_2] \begin{pmatrix} R_2 \\ S_2 \end{pmatrix} = d_1 \mathbf{A}^{-1} f \begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$. Therefore, we can evaluate $\hat{\psi}_1$ and $\hat{\psi}_2$ by **EvalByKani** with input

$$(m_1^2 d_1, m_2^2 d_2, F, E_1, E_2, [m_1]R_1, [m_1]S_1, [d_1]\mathbf{A}^{-1}(R_2, S_2)^\top).$$

Since the images of a basis of $E_1[d_1]$ under the composition $\hat{\phi}_{A,1} \circ \hat{\psi}_1$ generate $\ker \phi_1$, we can recover n_1 from $\mathbf{A}$ and $\phi_{A,1}$. Similarly, we can recover n_2 from $\mathbf{A}$ and $\phi_{A,2}$. Finally, we compute ϕ_1 from n_1 and recover $\mathbf{B}$ by computing the images of P_0 and Q_0 under ϕ_1.

Security. In [3], the following three problems are defined for discucsing the one-wayness of the FESTA trapdoor function. We say that a trapdoor function is a *one-way function* if there is no probabilistic polynomial-time algorithm to invert the function from public information and the output of the function with non-negligible probability.

Problem 1 (Decisional isogeny with scaled-torsion (DIST)). Let E_0 be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, and P_0, Q_0 be a basis of $E_0[n]$ for an integer n. Fix a degree d coprime with n, and given an elliptic curve E_1 and two points P_1, Q_1 sampled with probability 1/2 from either distribution:

– $\mathcal{D}_0 = (E_1, P_1, Q_1)$, where E_1 is the codomain of uniformly sampled d-isogeny $\phi : E_0 \to E_1$ and the points P_1, Q_1 are given by $(P_1, Q_1)^\top = \mathbf{A}(\phi(P_0), \phi(Q_0))^\top$, where the matrix $\mathbf{A} \in_U \mathcal{M}_n$,
– $\mathcal{D}_1 = (E_1, P_1, Q_1)$, where E_1 is a random elliptic curve over $\mathbb{F}_{p^2}$ with the same order of rational points as E_0, and (P_1, Q_1) is a random basis of $E_1[n]$,

distinguish from which distribution the values were sampled.

Problem 2 (Computational isogeny with scaled-torsion (CIST)). Let $\phi : E_0 \to E_1$ be an isogeny of smooth degree d between supersingular elliptic curves over $\mathbb{F}_{p^2}$, and let n be a smooth integer coprime with d. Given E_0, E_1, a basis P_0, Q_0 of $E_0[n]$, and $\mathbf{A}\phi(P_0, Q_0)^\top$, where $\mathbf{A} \in_U \mathcal{M}_n$, compute ϕ.

Problem 3 (Computational isogeny with double scaled-torsion (CIST ^{2})). Let E_0 and E_0' be two random supersingular elliptic curves over $\mathbb{F}_{p^2}$, and let $\phi_1 : E_0 \to E_1$ and $\phi_2 : E_0' \to E_2$ be two random isogenies of degrees d and d', respectively. Let n be an integer coprime with d, and let $\mathbf{A}$ be a matrix sampled as $\mathbf{A} \in_U \mathcal{M}_n$. Given the curves E_0, E_0', E_1, E_2, two bases $P, Q \in E_0[n]$ and $P', Q' \in E_0'$, and the points $\mathbf{A}(\phi_1(P), \phi_1(Q))^\top$ and $\mathbf{A}(\phi_2(P'), \phi_2(Q'))^\top$, compute the isogenies ϕ_1 and ϕ_2.

Remark 2. To compute an isogeny $\phi : E \to F$ as the answer of Problem 3 means to obtain a polynomial-time algorithm that takes an arbitrary point $P \in E$ as input and outputs $\phi(P)$. Note that we can compute ϕ_1 and ϕ_2 by executing the SIDH attack ([32], Section 2, dimension 8 attack) when we obtain the matrix $\mathbf{A}$.

Assuming the hardness of these problems for appropriate parameter, it is claimed that the FESTA trapdoor function is a one-way function [3, Theorem 9] and a quantum partial-domain one-way function [3, Theorem 10], i.e., it is hard to compute the first input n_1 in the input of the FESTA function from public information and the output.

IND-CCA secure KEM. Basso et al. [3] obtained an IND-CCA KEM by applying Optimal Asymmetric Encryption Padding (OAEP) transform [4] to the FESTA trapdoor function. Here, we briefly explain OAEP transform. For the detail of OAEP transform, see [4,17].

Let $F : \{0,1\}^{n+k_1} \times \{0,1\}^{k_0} \to \{0,1\}^m$ be a quantum partical-domain one-way function. Then we obtain a KEM with message space $\{0,1\}^n$ by applying OAEP transform to F. The obtained KEM is IND-CCA secure under QROM if $n + k_1 \geq k_0$ and $k_0 - n \approx n$ [17, Theorem 1]. Note that the bit length of a message of the obtained KEM is less than about one quarter of that of an input of the one-way function.

The bit length of an input of the FESTA trapdoor function is about $\log_2 d_1 + \log_2 d_2 + b \approx 7\lambda$. Therefore, by appropriately separating the domain of the FESTA trapdoor function and applying OAEP transform, we can obtain an IND-CCA secure KEM with sufficiently large message space.

3 QFESTA

This section introduces our protocol, a new PKE based on FESTA and some quaternion algebraic techniques. The original FESTA uses Vélu's formula [36] to compute the secret isogenies in **KeyGen** and **Enc**. Thus, their degrees must be smooth and divide $p+1$ to efficiently use Vélu's formula. This strong constraint makes p as large as $2^{8\lambda}$ for the security parameter λ, resulting in a large public key and ciphertext size.

Our main idea is to evaluate point images under isogenies of *non-smooth* degree not using Vélu's formula but using **FullRepresentInteger** and the SIDH attack. As a result, the size of the public key and ciphertext is nearly 1/3 of FESTA, though our protocol requires $(2,2)$-isogeny computations not only in **Dec** but also in **KeyGen** and **Enc**.

Our PKE protocol described here is OW-qPCA secure, and by applying $\mathbf{U}^{\not\perp}$ transform to the protocol, we obtain IND-CCA secure KEM. We name our new KEM 'QFESTA' (Quaternion Fast Encapsulation from Supersingular Torsion Attacks).

3.1 New Algorithm for Isogenies of Non-Smooth Degree

Here, we describe our new sub-algorithm **RandIsogImages** that evaluates point images under isogenies of *non-smooth* degree.

Let p be a prime such that $p \equiv 3 \mod 4$ and let E_0 be a supersingular elliptic curve defined as $E_0 : y^2 = x^3 + x/\mathbb{F}_{p^2}$. Note that $\mathrm{End}(E_0)$ is isomorphic to $\mathcal{O}_0 = \mathbb{Z}\langle 1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2}\rangle$ as mentioned in Section 2.3. Suppose that D is a smooth integer such that $E_0[D] \subset E_0(\mathbb{F}_{p^2})$ and $D \approx p$. Our sub-algorithm **RandIsogImages** takes a finite subset S of E_0 and an integer d coprime to D satisfying $d < D$ and $D - d \approx p$ as input. Then, it outputs the images $\tau(S)$ and the codomain E_A of a random isogeny τ of degree d.

The idea for this sub-algorithm is to compute an endomorphism $\alpha \in \mathrm{End}(E_0)$ of degree $d \cdot (D-d)$ by using **FullRepresentInteger**. Then, we can decompose α as $\alpha = \hat{\rho} \circ \tau$, where τ and ρ are the isogenies whose domains are E_0 and whose degree are d and $D - d$, respectively. (See Figure 2.) Since $\deg \tau + \deg \rho = D$ and $\gcd(\deg \tau, \deg \rho) = 1$, we can evaluate point images under the isogeny τ by using **CodomainByKani**. Especially when $D = 2^a$, we can compute it by using Richelot isogenies. So, we use $D = 2^a$ in our protocol. We describe the sub-algorithm in Algorithm 2.

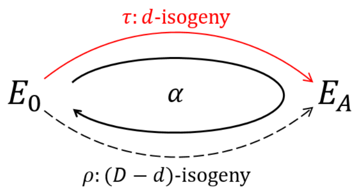

Fig. 2: Picture of **RandIsogImages**.

We discuss the output space of $\mathbf{RandIsogImages}_{\mathcal{O}_0}(S, d, D)$. We denote by $\mathbf{Cod}(E_0, S, d)$ the set of (E, S'), where E is the codomain of τ and $S' = \tau(S)$ for all d-isogenies τ from E_0. For any $(E, S') \in \mathbf{Cod}(E_0, S, d)$, there exits a $(D-d)$-isogeny $\rho : E_0 \to E$ in high probability since $D - d \approx p$. This is due to the Ramanujan property of the supersingular isogeny graphs [31]. Therefore, there exists an endomorphism $\alpha \in \mathrm{End}(E_0)$ via E of degree $d(D-d)$. When **FullRepresentInteger** outputs such α in step 1, **RandIsogImages** will output (E, S'). Though the output of **FullRepresentInteger** does not contain all endomorphisms of degree $d(D-d)$, we can assume that at least $1/\log(d(D-d))$ of all endomorphisms of degree $d(D-d)$ could be the output of **FullRepresentInteger** as mentioned in Section 2.3. Therefore, the output of **RandIsogImages** almost contains $\mathbf{Cod}(E_0, S, d)$. Since the number of d-isogenies from E_0 is about d, the number of possible outputs of **RandIsogImages** is about $d/\log(d(D-d))$. More precisely, we can assume that the probability of existing an isogeny ρ described above would be approximately $(D-d)/(p/12)$.

Algorithm 2 **RandIsogImages**$_{\mathcal{O}_0}(S, d, D)$

```
Require: Finite subset $S \subset E_0$, and integers $d, D$ such that $\gcd(d, D) = 1$, $d < D$,
    $D - d \approx p$, and $E_0[D] \subset E_0(\mathbb{F}_{p^2})$.
Ensure: $(\tau(S), E_A)$ for a random $d$-isogeny $\tau : E_0 \to E_A$.
1: Let $\alpha \leftarrow \mathbf{FullRepresentInteger}_{\mathcal{O}_0}(d \cdot (D - d))$.
2: Take a basis $P_0, Q_0$ of $E_0[D]$.
3: $(\tau(S), \emptyset, E_A) \leftarrow \mathbf{CodomainByKani}(d, D - d, E_0, E_0, P_0, Q_0, \alpha(P_0), \alpha(Q_0), S, \emptyset)$.
4: return $(\tau(S), E_A)$.
```

Therefore, we can assume that the number of output of **RandIsogImages** is approximately $(d/\log(d(D-d)))\cdot(D-d)/(p/12) = 12d(D-d)/p\log(d(D-d))$.

From the above argument, it seems possible to assume that the output distribution of **RandIsogImages** is indistinguishable from the distribution of the codomain and the point images of uniformly sampled d-isogeny from E_0. So, we assume the hardness of Problem 4.

Problem 4. Let E_0 be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, $\mathcal{O}_0 \cong \mathrm{End}(E_0)$, and P_0, Q_0 be a basis of $E_0[n]$ for an integer n. Fix integers d, D such that $\gcd(d, D) = 1$, $d < D$, $D - d \approx p$, and $E_0[D] \subset E_0(\mathbb{F}_{p^2})$. Given an elliptic curve E_1 and two points P_1, Q_1 sampled with probability $1/2$ from either distribution:

- $\mathcal{D}'_0 = (E_1, P_1, Q_1)$, the output of $\mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, d, D)$,
- $\mathcal{D}'_1 = (E_1, P_1, Q_1)$, where E_1 is the codomain of uniformly sampled d-isogeny $\phi : E_0 \to E_1$ and the points P_1, Q_1 are given by $(P_1, Q_1)^\top = (\phi(P_0), \phi(Q_0))^\top$,

distinguish from which distribution the values were sampled.

The hardness of Problem 1 and Problem 4 imply that of the following problem, which is a variant of Problem 1.

Problem 5 (A variant of DIST). Let E_0 be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, $\mathcal{O}_0 \cong \mathrm{End}(E_0)$, and P_0, Q_0 be a basis of $E_0[n]$ for an integer n. Fix integers d, D such that $\gcd(d, D) = 1$, $d < D$, $D - d \approx p$, and $E_0[D] \subset E_0(\mathbb{F}_{p^2})$. Given an elliptic curve E_1 and two points P_1, Q_1 sampled with probability $1/2$ from either distribution:

- $\mathcal{D}''_0 = (E_1, P_1, Q_1)$, where $(E_1, P, Q) \leftarrow \mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, d, D)$ and $(P_1, Q_1)^\top = \mathbf{A}(P, Q)^\top$, for a matrix $\mathbf{A} \in_U \mathcal{M}_n$,
- $\mathcal{D}''_1 = (E_1, P_1, Q_1)$, where E_1 is a random elliptic curve over $\mathbb{F}_{p^2}$ with the same order of rational points as E_0, and (P_1, Q_1) is a random basis of $E_1[n]$,

distinguish from which distribution the values were sampled.

3.2 PKE Protocol

Now, we describe our PKE protocol that is OW-qPCA secure. The proof of OW-qPCA security of our protocol is given in Section 4.1. The main difference of our protocol with FESTA is that we use *non-smooth* degree isogenies for ϕ_A and ϕ_1 in **KeyGen** and **Enc**. We show a picture of our protocol in Figure 3. As in Section 2.6, $\mathcal{M}_n$ represents the set of 2×2 diagonal regular matrices over $\mathbb{Z}/n\mathbb{Z}$. Our protocol is roughly outlined below:

- $\mathbf{Setup}(1^\lambda) \to \mathbf{param}$:
  1. Find integers p, a, b, k, D_A, and D_1 satisfying the following conditions:
     - $p = 2^a \cdot 3f - 1$ is a prime for a small integer f.
     - a and b are even integers satisfying $2^a \approx 3^b \approx 2^{2\lambda}$.
     - k is a small odd integer.
     - $D_A = 2^{a/2} + k \cdot 3^{b/2}$ and $D_1 = 2^{a/2} - k \cdot 3^{b/2}$ are positive integers containing a large prime factor larger than 2^λ. (We explain why D_A and D_1 should have large prime factors in Section 4.2.)
  2. Let $E_0 : y^2 = x^3 + x/\mathbb{F}_{p^2}$ and $\mathcal{O}_0 = \mathbb{Z}\langle 1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2}\rangle$.
  3. Take a basis (P_0, Q_0) of $E_0[2^a]$.
  4. Output a system parameter $\mathbf{param} = (p, a, b, k, D_A, D_1, E_0, \mathcal{O}_0, P_0, Q_0)$.
- $\mathbf{KeyGen}(\mathbf{param}) \to (pk, sk)$:
  1. Let $(\{P_A, Q_A\}, E_A) \leftarrow \mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, D_A, 2^a)$. (Denote the corresponding isogeny by ϕ_A.)
  2. Take a random diagonal matrix $\mathbf{A} \in_U \mathcal{M}_{2^a}$.
  3. Let $(R_A, S_A)^\top = \mathbf{A}(P_A, Q_A)^\top$.
  4. Output a public key $pk = (E_A, P_A, Q_A)$ and a secret key $sk = \mathbf{A}$.
- $\mathbf{Enc}(pk, m; \mathbf{param}) \to ct$:
  1. Convert the message m into a diagonal matrix $\mathbf{B} \in \mathcal{M}_{2^a}$.
  2. Let $(\{P_1, Q_1\}, E_1) \leftarrow \mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, D_1, 2^a)$. (Denote the corresponding isogeny by ϕ_1.)
  3. Let $(R_1, S_1)^\top = \mathbf{B}(P_1, Q_1)^\top$.
  4. Let $\phi_2 : E_A \to E_2$ be a random 3^b-isogeny and evaluate the points $(R_2, S_2)^\top = \mathbf{B}(\phi_2(R_A), \phi_2(S_A))^\top$.
  5. Output the ciphertext $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$.
- $\mathbf{Dec}(sk, ct; \mathbf{param}) \to m$:
  1. Let $\psi_1 = \phi_1 \circ \hat{\phi}_A$ and $\psi_2 = [k] \circ \phi_2$.
  2. Let $N_1 = \deg(\psi_1) = D_A D_1$ and $N_2 = \deg(\psi_2) = k^2 3^b$.
  3. Compute $(R_2', S_2') = (\psi_2 \circ \hat{\psi}_1(R_1), \psi_2 \circ \hat{\psi}_1(S_2))$.
  4. Execute $\mathbf{EvalByKani}(N_1, N_2, E_A, E_1, E_2, R_1, S_1, R_2', S_2', \emptyset, \{R_2, S_2\})$, and obtain $(R_{2,A}, S_{2,A}) = (\hat{\psi}_2(R_2), \hat{\psi}_2(S_2))$.
  5. Find $\mathbf{B} \in \mathcal{M}_{2^a}$ such that $(R_{2,A}, S_{2,A})^\top = k3^b\mathbf{B}(R_A, S_A)^\top$.
  6. Convert the matrix $\mathbf{B}$ to the message m.
  7. Confirm that there exists a $D_A D_1$-isogeny ψ_1 which is the composite of two isogenies: $\phi_1 \circ \hat{\phi}_A : E_A \to E_0 \to E_1$ of degree D_A and D_1. If there exits, output m, and otherwise, output $\perp$.

$$\begin{tikzcd}
\begin{pmatrix}P_0\\Q_0\end{pmatrix} & & \begin{pmatrix}R_A\\S_A\end{pmatrix} = \boldsymbol{A}\begin{pmatrix}\phi_A(P_0)\\\phi_A(Q_0)\end{pmatrix} \\
E_0 \arrow[rr, "\phi_A\colon D_A\text{-isogeny}"] \arrow[d, "\phi_1\colon D_1\text{-isogeny}"'] \arrow[d, dashed, shift left=2, "\psi_1"] & & E_A \arrow[d, "\phi_2\colon 3^b\text{-isogeny}"] \arrow[d, dashed, shift right=2, "\psi_2"'] \\
E_1 & & E_2 \\
\begin{pmatrix}R_1\\S_1\end{pmatrix} = \boldsymbol{B}\begin{pmatrix}\phi_1(P_0)\\\phi_1(Q_0)\end{pmatrix} & & \begin{pmatrix}R_2\\S_2\end{pmatrix} = \boldsymbol{B}\begin{pmatrix}\phi_2(R_A)\\\phi_2(S_A)\end{pmatrix}
\end{tikzcd}$$

Fig. 3: A picture of our protocol.

Step 7 of **Dec** is necessary to check a *validness* of the ciphertext ct, which is defined in Section 4.1. This check makes our protocol OW-qPCA secure (Theorem 3). Although we can achieve OW-CPA security without step 7, we need to use another Fujisaki-Okamoto transform $\mathbf{QFO}_m^{\not\perp}$ in [22] to convert OW-CPA PKE into IND-CCA KEM under QROM. Since this transform requires the execution of **Enc** in **Decaps**, the computational cost of the KEM rather increases. Step 7-10 of Algorithm 5 described below correspond to this step.

Remark 3. In our protocol, both parties should execute $\mathbf{RandIsogImages}_{\mathcal{O}_0}$. Thus, we need to assume the hardness of Problem 4 in addition to the hardness of Problem 1. Moreover, our protocol relies on both parties knowing $\mathcal{O}_0 \cong \mathrm{End}(E_0)$. As a result, we need to assume the hardness of Problem 1, Problem 4, and Problem 3 with E_0 restricted to the curve whose endomorphism ring is known.

Now, we describe the concrete algorithms for **KeyGen**, **Enc** and **Dec** in Algorithm 3, 4, and 5, respectively. We denote by QFESTA.PKE our PKE defined by these algorithms. As for **Setup**, we discuss in Section 3.3.

Note that we only use the diagonal matrix of determinant 1 since we can recover the determinant by using the 2^a-Weil pairing e as follows:

$$e(R_A, S_A) = e(P_0, Q_0)^{D_A \cdot \det \mathbf{A}}.$$

Note again that we can evaluate the point images $\{R_{A,2}, S_{A,2}\}$ in Algorithm 5 step 4 *up to the automorphism of* E_A. When $j(E_A) \neq 0, 1728$, the automorphism is $\{\pm 1\}$. Therefore, the matrix $\mathbf{B}$ is determined by $s_B \in (\mathbb{Z}/2^a\mathbb{Z})^*/\{\pm 1\}$. Since the following map

$$\eta : [0, 2^{a-2} - 1] \to (\mathbb{Z}/2^a\mathbb{Z})^*/\{\pm 1\},\ m \mapsto 2m + 1$$

is bijection, we choose $\{0, 1\}^{a-2}$ as the message space.

Algorithm 3 KeyGen(param)

```
Require: The system parameter $\mathbf{param} = (p, a, b, k, D_A, D_1, E_0, \mathcal{O}_0, P_0, Q_0)$.
Ensure: The key pair $(pk, sk)$.
1: Take a random matrix $\mathbf{A} \in_U \mathcal{M}_{2^a}$.
2: Let $(\{P_A, Q_A\}, E_A) \leftarrow \mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, D_A, 2^a)$.
3: Compute $(R_A, S_A)^\top = \mathbf{A}(P_A, Q_A)^\top$.
4: return $pk = (E_A, R_A, S_A)$ and $sk = \mathbf{A}$.
```

Algorithm 4 Enc(*pk*, *m*; param)

```
Require: The public key $pk = (E_A, R_A, S_A)$, the message $m \in \{0,1\}^{a-2}$, and the
    system parameter $\mathbf{param}$.
Ensure: The ciphertext $ct$.
1: Let $s_B = 2m + 1 \in (\mathbb{Z}/2^a\mathbb{Z})^*$ and $\mathbf{B} = \mathrm{diag}(s_B, s_B^{-1}) \in \mathcal{M}_{2^a}$.
2: Let $(\{P_1, Q_1\}, E_1) \leftarrow \mathbf{RandIsogImages}_{\mathcal{O}_0}(\{P_0, Q_0\}, D_1, 2^a)$.
3: Compute $(R_1, S_1)^\top = \mathbf{B}(P_1, Q_1)^\top$.
4: Take a random $3^b$-isogeny $\phi_2 : E_A \to E_2$.
5: Compute $(R_2, S_2)^\top = \mathbf{B}(\phi_2(R_A), \phi_2(S_A))^\top$.
6: return $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$.
```

Correctness. We show the correctness of our PKE. In step 1-3 of Algorithm 5, we used Theorem 1 for $\psi_1 = \phi_1 \circ \hat{\phi}_A$, $\psi_2 = [k] \circ \phi_2$, $N_1 = D_A D_1$, $N_2 = k^2 3^b$, and $f = \psi_2 \circ \hat{\psi}_1$. Here, we have $N_1 + N_2 = D_A D_1 + k^2 3^b = 2^a$, and the following equation holds:

$$\begin{aligned}
(f(R_1), f(S_1))^\top &= ([k] \circ \phi_2 \circ \phi_A \circ \hat{\phi}_1(R_1), [k] \circ \phi_2 \circ \phi_A \circ \hat{\phi}_1(S_1))^\top \\
&= kD_1\mathbf{B}(\phi_2 \circ \phi_A(P_0), \phi_2 \circ \phi_A(Q_0))^\top \\
&= kD_1\mathbf{B}\mathbf{A}^{-1}(\phi_2(R_A), \phi_2(S_A))^\top \\
&= kD_1\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{-1}(R_2, S_2)^\top \\
&= kD_1\mathbf{A}^{-1}(R_2, S_2)^\top \\
&= (R_2', S_2')^\top.
\end{aligned}$$

Since the orders of R_1, S_1, R_2', and S_2' are all $N_1 + N_2 = D_A D_1 + k^2 3^b = 2^a$, **EvalByKani** in step 3 succeed if the ciphertext ct is generated honestly.

Algorithm 5 $\mathbf{Dec}(sk, ct; \mathbf{param})$

```
Require: The secret key $sk = s_A$, the ciphertext $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$, and
    the system parameter $\mathbf{param}$.
Ensure: The decrypted message $m$.
 1: Let $N_1 = D_A D_1$ and $N_2 = k^2 3^b$.
 2: Compute $(R_2', S_2')^\top = kD_1\mathbf{A}^{-1}(R_2, S_2)^\top$.
 3: $(\emptyset, \{R_{2,A}, S_{2,A}\}) \leftarrow$ EvalByKani$(N_1, N_2, E_A, E_1, E_2, R_1, S_1, R_2', S_2', \emptyset, \{R_2, S_2\})$.
 4: if EvalByKani returns $\perp$, return $\perp$.
 5: Find $\mathbf{B} = \mathrm{diag}(s_B, s_B^{-1}) \in \mathcal{M}_{2^a}$ such that $(R_{2,A}, S_{2,A})^\top = k3^b\mathbf{B}(R_A, S_A)^\top$ by
    solving discrete logarithm problem.
 6: if there is no such $s_B \in (\mathbb{Z}/2^a\mathbb{Z})^*/\pm 1$, return $\perp$.
 7: Let $(R_1', S_1')^\top = 2^{a/2-1}\mathbf{B}^{-1}(R_1, S_1)^\top$.
 8: Let $(R_A', S_A')^\top = 2^{a/2-1}D_A^{-1}\mathbf{A}^{-1}(R_A, S_A)^\top$.
 9: Execute EvalByKani$(D_A, D_1, E_0, E_A, E_1, R_A', S_A', R_1', S_1', \emptyset, \emptyset)$.
10: if EvalByKani returns $\perp$, return $\perp$.
11: Let $s_B \leftarrow \min\{s_B, 2^a - s_B\}$.
12: return $m = (s_B - 1)/2$.
```

In step 7-9, we again used Theorem 1 for $\psi_1 = \phi_A$, $\psi_2 = \phi_1$, $N_1 = D_A$, $N_2 = D_1$, and $f = \phi_1 \circ \hat{\phi_A}$. Here, the following equation holds:

$$\begin{aligned}(f(R_A'), f(S_A'))^\top &= (\phi_1 \circ \hat{\phi_A}(R_A'), \phi_1 \circ \hat{\phi_A}(S_A'))^\top \\ &= 2^{a/2-1}D_A^{-1}\mathbf{A}^{-1}(\phi_1 \circ \hat{\phi_A}(R_A), \phi_1 \circ \hat{\phi_A}(S_A))^\top \\ &= 2^{a/2-1}(\phi_1(P_0), \phi_1(Q_0))^\top \\ &= (R_1', S_1')^\top.\end{aligned}$$

Since the orders of R_1', S_1', R_A', and S_A' are all $N_1 + N_2 = D_A + D_1 = 2^{a/2+1}$, **EvalByKani** in step 9 succeed if the ciphertext ct is generated honestly. From the above discussion, the correctness of our protocol follows.

Remark 4. We can efficiently compute the 3^b-isogeny in Step 5 of Algorithm 4 by using the radical isogenies [8]. It should be noted, however, that we require the point images, unlike the original radical isogenies. We show the method in Appendix A.

3.3 Parameter Finding

Here, we show how to find a system parameter $\mathbf{param} = (p, a, b, k, D_A, D_1, E_0, \mathcal{O}_0, P_0, Q_0)$ for a given security parameter λ. The discussion of parameter sizes is provided in Section 4.2.

First, we take $a = 2a', b = 2b' \in 2\mathbb{N}$ such that $2^a \approx 3^b \approx 2^{2\lambda}$ and then find $k \in \mathbb{N}$ such that both $D_A = 2^{a'} + k3^{b'}$ and $D_1 = 2^{a'} - k3^{b'}$ have a prime factor larger than 2^λ by trying k in ascending order. If D_1 becomes negative, increment a' and try k again from 1. Next, we find $f \in \mathbb{N}$ such that $p = 2^a \cdot 3f - 1$ is prime. We can try f in ascending order until $p = 2^a \cdot 3f - 1$ becomes prime. Finally, we set E_0 and $\mathcal{O}_0$ as $E_0 : y^2 = x^3 + x/\mathbb{F}_{p^2}$ and $\mathcal{O}_0 = \mathbb{Z}\langle 1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2}\rangle$, respectively, and we find a basis (P_0, Q_0) of $E_0[2^a]$. We show the algorithm for **Setup** in Algorithm 6.

Algorithm 6 Setup(1^λ)

```
Require: The security parameter $1^\lambda$.
Ensure: The system parameter **param**.
 1: Let $S = \{$Set of available integers for $f\}$.
 2: Let $a' = \lambda$ and $b' = \lceil \lambda \log_3 2 \rceil$.
 3: while true do
 4:    Let $k = 1$.
 5:    while $D_1 = 2^{a'} - k3^{b'} > 0$ do
 6:       Let $D_A = 2^{a'} + k3^{b'}$.
 7:       if $D_A$ and $D_1$ have a prime factor larger than $2^\lambda$ then
 8:          for $f \in S$ (ascending order) do
 9:             if $p = 2^{2a'} \cdot 3f - 1$ is prime then
10:                Go to Step 18.
11:             end if
12:          end for
13:       end if
14:       Let $k = k + 1$.
15:    end while
16:    Let $a' = a' + 1$.
17: end while
18: Let $a = 2a'$ and $b = 2b'$.
19: Let $E_0 : y^2 = x^3 + x/\mathbb{F}_{p^2}$ and $\mathcal{O}_0 = \mathbb{Z}\langle 1, \mathbf{i}, \frac{\mathbf{i}+\mathbf{j}}{2}, \frac{1+\mathbf{k}}{2}\rangle$.
20: Take a basis $(P_0, Q_0)$ of $E_0[2^a]$.
21: return param $= (p, a, b, k, D_A, D_1, E_0, \mathcal{O}_0, P_0, Q_0)$
```

Remark 5. We need to compute cube roots of elements in $\mathbb{F}_{p^2}$ for the radical isogenies. In the case that $p + 1$ is divisible by 3 and not by 9, the cube root computation is efficient. Therefore, it is preferable that f is not divisible by 3. For more detail, see Appendix A.2. Additionally, it is preferable that f is odd since we use 2^a-Tate pairing to solve the discrete logarithm problem in Algorithm 5. Thus, the set S in step 1 could be $S = \{f \in \mathbb{N} \mid \gcd(f, 6) = 1, f \leq B_f\}$ for a bound

B_f. From the prime number theory, we can find f around $O(\log 2^a) = O(\lambda)$. Therefore, we can choose the bound B_f in $O(\lambda)$. Note that the requirement of $\gcd(f, 6) = 1$ is just a preferred condition for implementation, not a theoretical requirement.

4 Security Analysis

In this section, we analyse the security of QFESTA. The authors of FESTA applied OAEP to their protocol to achieve IND-CCA security. Our protocol, however, is not suitable for OAEP. The reason is as follows. Our protocol can be seen as a one-way trapdoor function with the domain $\{0,1\}^{a-2}$. Therefore, by applying OAEP, the massage size will be about $a/4$ bits, which is about $\lambda/2$ bits since $a \approx 2\lambda$. This message size is too small to achieve the λ-bit security. Instead, we use the Fujisaki-Okamoto transform [20] to achieve IND-CCA security.

4.1 Security Proof

In this subsection, we prove that our PKE is OW-qPCA secure; thus, our QFESTA is IND-CCA secure. First, we define a notion to simplify our proof. Next, we show that the adversary can simulate the plaintext checking oracle **Pco** defined in Definition 3 by executing a polynomial-time algorithm. This fact allows us to reduce the proof of OW-qPCA security to the proof of one-wayness with no oracle. Thus, we can easily prove that our PKE is OW-qPCA secure. Finally, we prove that our QFESTA is IND-CCA secure by applying the Fujisaki-Okamoto transform.

Throughout this subsection, we fix a system parameter **param** and a key pair (pk, sk) arbitrarily. For simplicity, we denote by $\mathbf{B}(m)$ the diagonal matrix corresponding to the message m. First, we define the validness of the ciphertext of our PKE, which roughly means that the ciphertext is generated honestly.

Definition 5. *Let E_1 and E_2 be supersingular elliptic curves over $\mathbb{F}_{p^2}$ and let (R_1, S_1) and (R_2, S_2) be basis of $E_1[2^a]$ and $E_2[2^a]$, respectively. For a message $m \in \{0,1\}^{a-2}$, we say that the ciphertext $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$ is m-*valid *if it satisfies the following conditions:*

– *There exists a D_1-isogeny $\phi_1 : E_0 \to E_1$ and a $k^2 3^b$-isogeny $\psi_2 : E_A \to E_2$ such that*

$$(R_1, S_1)^\top = \mathbf{B}(m)(\phi_1(P_0), \phi_1(Q_0))^\top,$$
$$k(R_2, S_2)^\top = \mathbf{B}(m)(\psi_2(R_A), \psi_2(S_A))^\top.$$

Next, we prove that our **Dec** algorithm accepts only the valid ciphertext.

Lemma 1. *Let E_1 and E_2 be random elliptic curves isogenous to E_0 and let (R_1, S_1) and (R_2, S_2) be random basis of $E_1[2^a]$ and $E_2[2^a]$, respectively. Then, the following two statements are equivalent with all but negligible probability.*

(a) *The ciphertext $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$ is m-valid.*
(b) $\mathbf{Dec}(sk, ct) = m$.

Proof. We can assume that $j(E_1) \neq 0, 1728$ and $j(E_2) \neq 0, 1728$ with all but negligible probability. From the correctness of our PKE, it is apparent that $\mathbf{Dec}(sk, ct)$ returns m if ct is m-valid. Conversely, we assume $\mathbf{Dec}(sk, ct) = m$. The probability of **EvalByKani** succeeding for inputs that do not satisfy the conditions of Theorem 1 is negligible. Therefore, the inputs of **EvalByKani** satisfy the conditions of Theorem 1 with all but negligible probability when **EvalByKani** does not output $\perp$. Then, $k(R_2, S_2)^\top = \mathbf{B}(m)(\psi_2(R_A), \psi_2(S_A))^\top$ holds for a $k^2 3^b$-isogeny ψ_2 since step 4 and 6 of Algorithm 5 does not return $\perp$. Similarly, $(R_1, S_1)^\top = \mathbf{B}(m)(\phi_1(P_0), \phi_1(Q_0))^\top$ holds for a D_1-isogeny ϕ_1 since step 10 of Algorithm 5 does not return $\perp$. Therefore, the ciphertext ct is m-valid. □

Then, we can prove the following lemma, which indicates that the adversary can simulate the oracle **Pco** in polynomial time.

Lemma 2. *Let $\widehat{\mathbf{Pco}}(m, ct)$ be the following polynomial-time algorithm:*

1. *Parse $ct = (E_1, R_1, S_1, E_2, R_2, S_2)$.*
2. *Let $(P_1, Q_1)^\top = \mathbf{B}(m)^{-1}(R_1, S_1)^\top$.*
3. *Compute a D_1-isogeny $\phi_1 : E_0 \to E_1$ such that $\phi_1(P_0) = P_1$ and $\phi_1(Q_0) = Q_1$ by the SIDH attack ([32], Section 2, dimension 8 attack). If the attack failed, return* **false**.
4. *Let $(P_2, Q_2)^\top = k\mathbf{B}(m)^{-1}(R_2, S_2)^\top$.*
5. *Compute a $k^2 3^b$-isogeny $\psi_2 : E_A \to E_2$ such that $\psi_2(R_A) = P_2$ and $\psi_2(S_A) = Q_2$ by the SIDH attack ([32], Section 2, dimension 8 attack). If the attack failed, return* **false**.
6. *If both attacks succeeded, return* **true**.

Then, $\widehat{\mathbf{Pco}}(m, ct) = \mathbf{Pco}(m, ct)$ holds with all but negligible probability.

Proof. Assume that $j(E_1) \neq 0, 1728$ and $j(E_2) \neq 0, 1728$. It holds with all but negligible probability. If $\widehat{\mathbf{Pco}}(m, ct) = 1$, then ct is m-valid from the definition of $\widehat{\mathbf{Pco}}$. Then, $\mathbf{Pco}(m, ct) = 1$ from Lemma 1. Conversely, if $\mathbf{Pco}(m, ct) = 1$, then ct is m-valid from Lemma 1. Therefore, $\widehat{\mathbf{Pco}}(m, ct) = 1$ from the definition of m-validness. □

Now, we prove that our PKE is OW-qPCA secure.

Theorem 3. *QFESTA.PKE is OW-qPCA secure under the following assumption.*

Assumption 1 *Problem 1 and Problem 4 is hard for $j(E_0) = 1728$, $d = D_A, D_1$, and $D = n = 2^a$ and Problem 3 is also hard for $j(E_0) = 1728$, $d_1 = D_1$, $d_2 = 3^b$, and $n = 2^a$.*

Proof. From Lemma 2, the adversary can simulate the oracle **Pco** by executing $\widehat{\mathbf{Pco}}$ with quantum state. Therefore, it is sufficient to prove that QFESTA.PKE is one-way.

Now, we prove the one-wayness of QFESTA.PKE. Assume that there exists an adversary **Adv** that breaks the one-wayness of QFESTA.PKE. Note that we can assume the hardness of Problem 5 from the hardness of Problem 1 and Problem 4. Let (E_A, R_A, S_A) be an input of Problem 5 for $j(E_0) = 1728$, $d = D_A$, and $D = n = 2^a$ and let (E_1, P_1, Q_1) be an input of Problem 4 for $j(E_0) = 1728$, $d = D_1$, and $D = n = 2^a$. Next, we take $\mathbf{B} \in_U \mathcal{M}_{2^a}$ and a random 3^b-isogeny $\phi_2 : E_A \to E_2$, let $(R_1, S_1)^\top = \mathbf{B}(P_1, Q_1)^\top$, and let $(R_2, S_2)^\top = \mathbf{B}(\phi_2(R_A), \phi_2(S_A))^\top$. From the hardness of Problem 4 and Problem 5, $\mathbf{Adv}(E_1, R_1, S_1, E_2, R_2, S_2)$ will output $\mathbf{B}$ regardless of whether (E_A, R_A, S_A) is sampled from $\mathcal{D}''_0$ or $\mathcal{D}''_1$ and whether (E_1, P_1, Q_1) is sampled from $\mathcal{D}'_0$ or $\mathcal{D}'_1$. Therefore, $\mathbf{Adv}(E_1, R_1, S_1, E_2, R_2, S_2)$ will output $\mathbf{B}$ even when $(E_A, R_A, S_A) \in \mathcal{D}''_0$ and $(E_1, P_1, Q_1) \in \mathcal{D}'_0$, which means that **Adv** can solve Problem 3 for $j(E_0) = 1728$, $d_1 = D_1$, $d_2 = 3^b$, and $n = 2^a$. This is contrary to our assumption. Therefore, QFESTA.PKE is one-way. □

Consequently, the following theorem immediately follows from Theorem 2 and Theorem 3.

Theorem 4. *Let QFESTA be the KEM obtained by applying the Fujisaki-Okamoto transform* $\mathbf{U}^{\not\perp}$ *to QFESTA.PKE. Then, QFESTA is IND-CCA KEM under QROM under Assumption 1.*

4.2 Hardness Analysis

Here, we discuss possible attacks against QFESTA and confirm that the parameters we have presented are of sufficient size to achieve λ-bit security.

In our protocol, we primarily publish three types of information: elliptic curves E_0, E_A, E_1, E_2, masked torsion points $P_A, Q_A, R_1, S_1, R_2, S_2$, and the degrees $D_A, D_1, 3^b$ of each isogeny. To obtain the plaintext m in our protocol, it is necessary and sufficient to compute one of the three secret isogenies, namely, ϕ_A, ϕ_1, or ϕ_2.

An efficient method for computing isogenies using masked torsion points was introduced in [9]. However, we can avoid this attack in the way described in Remark 1. Apart from this method, there is no known efficient method to compute isogenies using masked torsion points.

Now, we focus on the problem of finding the isogeny $\phi_A : E_0 \to E_A$, $\phi_1 : E_0 \to E_1$, or $\phi_2 : E_A \to E_2$ when given elliptic curves E_0, E_A, E_1, E_2 and each degree $D_A, D_1, 3^b$. Three attack methods are considered: (i) exhaustive search of all outputs of **RandIsogImages**, (ii) meet-in-the-middle strategies [1], and (iii) computing the endomorphism ring of the elliptic curve.

(i) The number of possible outputs of $\mathbf{RandIsogImages}_{\mathcal{O}_0}(S, D_A, 2^a)$ is approximately $12D_A(2^a - D_A)/p \log D_A(2^a - D_A)$ as we explained in Section 3.1. In our setting, we have

$$12D_A(2^a - D_A)/p \log D_A(2^a - D_A) \approx 4D_A/3\lambda f \in \tilde{O}(2^\lambda)$$

since $D_A \approx 2^\lambda$, $2^a \approx 2^{2\lambda}$, and $p = 2^a \cdot 3f - 1$. Consequently, the computational cost for this attack is $\tilde{O}(2^\lambda)$. In the case of a quantum adversary, Grover's algorithm reduces the cost to $\tilde{O}(2^{\lambda/2})$. The same argument as above follows for D_1. To be precise, $4D_A/3\lambda f$ is slightly smaller than 2^λ, but this is not considered to have a significant impact on the security of QFESTA. More detailed analysis for this is left for a future work.

(ii) We discuss the computational cost of meet-in-the-middle strategies against a d-isogeny from E to F. When the degree d can be factored as $d = d' \cdot d''$, we search exhaustively for d'-isogenies from E and d''-isogenies from F. This results in a computational cost of $O(d' + d'')$. In our setting, both D_A and D_1 have a prime factor greater than 2^λ, meaning either d' or d'' becomes larger than 2^λ. Therefore, the attack's cost remains $O(2^\lambda)$. On the other hand, as for $d = 3^b$, we have $d' = d'' = 3^{b/2} \approx 2^\lambda$, leading to an attack cost of $O(2^\lambda)$. In the case of a quantum adversary, by using the method in [35], the cost is reduced to $O(2^{2\lambda/3})$.

(iii) By using the method described in Section 4.2 of [21], we can compute ϕ_A in polynomial time from the endomorphism ring $\mathrm{End}(E_A)$. This is because the degree D_A of ϕ_A is approximately $\sqrt{p}$, meaning that ϕ_A has the smallest degree among the isogenies between E_0 and E_A in high probability. Now, we discuss the way of computing the endomorphism ring $\mathrm{End}(E_A)$. When we find an isogeny between E_0 and E_A of arbitrary degree by executing Delfs-Galbraith attack [15], we can compute $\mathrm{End}(E_A)$ in polynomial time [18]. The cost for Delfs-Galbraith attack is $\tilde{O}(p^{1/2}) = \tilde{O}(2^\lambda)$ for a classical adversary. The endomorphism ring $\mathrm{End}(E_A)$ can also be obtained directly by the method in [19] and the cost is also $\tilde{O}(p^{1/2}) = \tilde{O}(2^\lambda)$ for a classical adversary. In these attacks, the most computationally intensive task involves searching for a path in the supersingular isogeny graph to a curve within a specific set, which has an approximate cardinality of $O(p^{1/2})$. Therefore, a quantum adversary has the potential to reduce the computational costs of these attacks to $O(p^{1/4}) = O(2^{\lambda/2})$ using Grover's algorithm.

From the above discussion, it is likely that our parameter settings afford λ-bit security against a classical adversary and $\lambda/2$-bit security against a quantum adversary.

5 Efficiency

In this section, we analyse the efficiency of QFESTA. First, we provide concrete parameters for QFESTA, then compare the data sizes of QFESTA such as public key size and ciphertext size with FESTA. Finally, we show the computational cost by our proof-of-concept implementation.

5.1 Parameter

This subsection gives concrete parameters for QFESTA satisfying the NIST security level 1, 3, and 5. The parameters are generated by Algorithm 6 while we take the set S in step 1 as $S = \{f \in \mathbb{N} \mid \gcd(f, 6) = 1,\ f \leq 1000\}$ (see Remark 5) and the security parameter as $\lambda = 128, 192, 256$, respectively. We denote QFESTA with the parameter for NIST security level 1, 3, and 5 by 'QFESTA-128', 'QFESTA-192', and 'QFESTA-256', respectively. The parameters are as follows:

– QFESTA-128:

$$\begin{aligned} p &= 2^{272} \cdot 3 \cdot 169 - 1,\ a = 272,\ b = 162,\ k = 131, \\ D_A &= 7 \cdot 41462022817031855404861711273462894536 49, \\ D_1 &= 19 \cdot 76421660995577997110444526901916472678 91. \end{aligned}$$

– QFESTA-192:

$$\begin{aligned} p &= 2^{404} \cdot 3 \cdot 671 - 1,\ a = 404,\ b = 244,\ k = 183, \\ D_A &= 7 \cdot 31 \cdot 15981918032612409671047655254204289906226136481918 57 \\ &\quad 4050521, \\ D_1 &= 93874281409950293057183555485669699105265523336860121134 47 \\ &\quad 951. \end{aligned}$$

– QFESTA-256:

$$\begin{aligned} p &= 2^{530} \cdot 3 \cdot 7 - 1,\ a = 530,\ b = 324,\ k = 15, \\ D_A &= 13 \cdot 37 \cdot 1171229603583629903380399286830447548370637845834 61 \\ &\quad 013065057280205783838893337, \\ D_1 &= 23 \cdot 2705867628114747207006064476189212633266121039694399577 \\ &\quad 9179136628313486192869 29. \end{aligned}$$

5.2 Data Size

In this subsection, we compare the data sizes of FESTA and QFESTA using the above parameters. The parameter of FESTA-128 is given in Section 7.3 of [3]. As for the parameters of FESTA-192 and FESTA-256, we used the values given in the FESTA implementation at: `https://github.com/FESTA-PKE/FESTA-SageMath`. Note that we used the latest version of FESTA at this time (updated August 19th, 2023).

Now, we compare the sizes of characteristic p, public key, and ciphertext of SIKE [2], FESTA [3], and QFESTA in Table 2. Note that all public key and ciphertext sizes in the table are values using key compression, as in SIKE. As shown in Table 2, all the data sizes of our protocol are much smaller than those of FESTA. In particular, the public key and ciphertext sizes of QFESTA-128 are nearly three to four times smaller than those of FESTA-128. Moreover, our public key size is smaller than SIKE.

Security	Protocol	p (bits)	Public key (bytes)	Ciphertext (bytes)
Level 1	SIKEp434	434	197	236
	FESTA-128	1292	561	1122
	QFESTA-128	**281**	**174**	**348**
Level 3	SIKEp610	610	274	336
	FESTA-192	1966	864	1728
	QFESTA-192	**415**	**257**	**514**
Level 5	SIKEp751	751	335	410
	FESTA-256	2772	1246	2492
	QFESTA-256	**535**	**335**	**670**

Table 2: Data size comparison

5.3 Implementation

We provide a proof-of-concept implementation of QFESTA in SageMath [34] and make it available at: `https://github.com/hiroshi-onuki/QFESTA-SageMath`. Now, we show the number of isogeny computations required for FESTA and QFESTA in Table 3 to compare the computational cost. As shown in Table 3, our protocol does not require high-degree isogeny computations for **KeyGen** and **Enc**, whereas FESTA requires a lot. In particular, the higher the security level, FESTA requires higher-degree isogeny computations, making QFESTA more advantageous. As for **Dec**, our protocol requires less $(2, 2)$-isogeny computations than FESTA. Furthermore, since our protocol uses a smaller p, the computational cost would be even lower.

Finally, in Table 4, we show the actual computational times of FESTA and QFESTA implemented in SageMath. These are the averages of 10 run times. Our running environment is an Apple M1 CPU (3.2 GHz). Note that these comparisons are not rigorous since both FESTA and QFESTA implementations are just proof-of-concept. Optimized implementation of QFESTA in C or other languages is a future work.

6 Conclusion

In this paper, we introduce QFESTA, a new variant of FESTA that works with better parameters. The main idea of our protocol is to compute a non-smooth degree isogeny by using **FullRepresentInteger** and the SIDH attack. The removal of the smoothness restriction allows us to use more efficient parameters.

Indeed, the data sizes of the public key and ciphertext of QFESTA become nearly three to four times smaller than those of FESTA in NIST security level 1, 3, and 5. Additionally, QFESTA is expected to have less computational cost since it only requires $(2, 2)$-isogeny and 3-isogeny computations, whereas the original FESTA requires high-degree isogeny computations. Especially as the security level increases, the advantages of QFESTA expand.

As a future work, we need to analyse the number of possible outputs of Algorithm 2 for a concrete parameter and its effect on the security. An optimized

Protocol		$(2,2)$	3	high-degree
FESTA-128	**KeyGen**	-	-	22 (degree: 59-41161)
	Enc	-	6	69 (degree: 5-3779)
	Dec	632	-	-
QFESTA-128	**KeyGen**	272	-	-
	Encaps	272	162	-
	Decaps	409	-	-
FESTA-192	**KeyGen**	-	-	22 (degree: 31-6842881)
	Enc	-	5	79 (degree: 5-176549)
	Dec	992	-	-
QFESTA-192	**KeyGen**	404	-	-
	Encaps	404	244	-
	Decaps	607	-	-
FESTA-256	**KeyGen**	-	-	26 (degree: 2729-44988859)
	Enc	-	4	105 (degree: 5-513031)
	Dec	1472	-	-
QFESTA-256	**KeyGen**	530	-	-
	Encaps	530	324	-
	Decaps	796	-	-

Table 3: Number of isogeny computations of each degree

Protocol	**KeyGen**	**Enc/Encaps**	**Dec/Decaps**
FESTA-128	4.88	3.13	9.25
QFESTA-128	**1.61**	**1.85**	**2.70**
FESTA-192	103.34	20.90	24.58
QFESTA-192	**2.77**	**3.46**	**5.63**
FESTA-256	298.06	58.06	58.06
QFESTA-256	**4.51**	**5.81**	**10.16**

Table 4: Computational times (sec.)

implementation of QFESTA is also a future work. In particular, since the Richelot isogeny computations accounts for about 80% of the total computational cost, speeding up the Richelot isogeny computation is an important future work.

Acknowledgments

We would like to thank Katsuyuki Takashima and Tomoki Moriya for the many suggestions for improving our paper. We also thank Keita Xagawa for providing valuable information on Fujisaki-Okamoto transform under QROM.

References

1. Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. In *Selected Areas in Cryptography*, pages 322–343. Springer, 2018.
2. Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, David Jao, Brian Koziel, Brian LaMacchia, Patrick Longa, et al. Supersingular isogeny key encapsulation. *Submission to the NIST Post-Quantum Standardization project*, 152:154–155, 2017.
3. Andrea Basso, Luciano Maino, and Giacomo Pope. FESTA: Fast encryption from supersingular torsion attacks. *Cryptology ePrint Archive, 2023/660*, 2023.
4. Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *EUROCRYPT 1994: Workshop on the Theory and Application of Cryptographic Techniques*, pages 92–111. Springer, 1995.
5. Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. In *ANTS-XIV - 14th Algorithmic Number Theory Symposium*, volume 4 of *Proceedings of the Fourteenth Algorithmic Number Theory Symposium (ANTS-XIV)*, pages 39–55. Mathematical Sciences Publishers, 2020.
6. Fouazou Lontouo Perez Broon, Thinh Dang, Emmanuel Fouotsa, and Dustin Moody. Isogenies on twisted Hessian curves. *Journal of Mathematical Cryptology*, 15(1):345–358, 2021.
7. Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In *EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 423–447. Springer, 2023.
8. Wouter Castryck, Thomas Decru, and Frederik Vercauteren. Radical isogenies. In *ASIACRYPT 2020: International Conference on the Theory and Application of Cryptology and Information Security*, pages 493–519. Springer, 2020.
9. Wouter Castryck and Frederik Vercauteren. A polynomial-time attack on instances of M-SIDH and FESTA. *Cryptology ePrint Archive, 2023/1433*, 2023.
10. Jorge Chavez-Saab, Maria Corte-Real Santos, Luca De Feo, Jonathan Komada Eriksen, Basil Hess, David Kohel, Antonin Leroux, Patrick Longa, Michael Meyer, Lorenz Panny, Sikhar Patranabis, Christophe Petit, Francisco Rodríguez Henríquez, Sina Schaeffler, and Benjamin Wesolowski. SQIsign. Submission to NIST standardization of additional digital signature schemes. `https://sqisign.org`, 2023.
11. Romain Cosset and Damien Robert. Computing (l, l)-isogenies in polynomial time on Jacobians of genus 2 curves. *Mathematics of Computation*, 84(294):1953–1975, 2015.
12. Richard Crandall and Carl B. Pomerance. *Prime Numbers: A Computational Perspective.* Springer, second edition, 2005.
13. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In *ASIACRYPT 2020: International Conference on the Theory and Application of Cryptology and Information Security*, pages 64–93. Springer, 2020.
14. Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the Deuring correspondence. In *EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 659–690. Springer, 2023.

15. Christina Delfs and Steven D Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *Designs, Codes and Cryptography*, 78:425–440, 2016.
16. Max Deuring. Die typen der multiplikatorenringe elliptischer funktionenkörper. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 14:197–272, 1941.
17. Ehsan Ebrahimi. Post-quantum security of plain OAEP transform. In *PKC 2022: IACR International Conference on Public-Key Cryptography*, pages 34–51. Springer, 2022.
18. Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: reductions and solutions. In *EUROCRYPT 2018: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 329–368. Springer, 2018.
19. Kirsten Eisenträger, Sean Hallgren, Chris Leonardi, Travis Morrison, and Jennifer Park. Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. *Open Book Series*, 4(1):215–232, 2020.
20. Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In *CRYPTO 2001: Annual International Cryptology Conference*, pages 260–274. Springer, 2001.
21. Steven D Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In *ASIACRYPT 2016: International Conference on the Theory and Application of Cryptology and Information Security*, pages 63–91. Springer, 2016.
22. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In *Theory of Cryptography*, pages 341–371. Springer, 2017.
23. Everett W. Howe, Franck Leprévost, and Bjorn Poonen. Large torsion subgroups of split Jacobians of curves of genus two or three. *Forum Mathematicum*, 12(3):315–364, 2000.
24. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *PQCrypto 2011: International Workshop on Post-Quantum Cryptography*, pages 19–34. Springer, 2011.
25. Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In *PQCrypto 2019: International Workshop on Post-Quantum Cryptography*, pages 227–248. Springer, 2019.
26. Ernst Kani. The number of curves of genus two with elliptic differentials. 1997.
27. David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17(A):418–432, 2014.
28. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In *EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 448–471. Springer, 2023.
29. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery on SIDH. *EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 448–471, 2023.
30. Hiroshi Onuki and Tomoki Moriya. Radical isogenies on Montgomery curves. In *PKC 2022: IACR International Conference on Public-Key Cryptography*, pages 473–497. Springer, 2022.

31. Arnold K. Pizer. Ramanujan graphs and Hecke operators. *Bulletin of the American Mathematical Society*, 23(1):127–137, 1990.
32. Damien Robert. Breaking SIDH in polynomial time. In *EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 472–503. Springer, 2023.
33. Benjamin Andrew Smith. *Explicit endomorphisms and correspondences.* Phd thesis, University of Sydney, 2005.
34. W. A. Stein et al. *Sage Mathematics Software (Version 9.8).* The Sage Development Team, 2023. `http://www.sagemath.org`.
35. Seiichiro Tani. Claw finding algorithms using quantum walk. *Theoretical Computer Science*, 410(50):5285–5297, 2009.
36. Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes-Rendus de l'Académie des Sciences*, 273:238–241, 1971.

A Computing a chain of 3-isogenies

In this section, we give an explicit algorithm to compute a chain of 3-isogenies in the encryption of our protocol.

Let p be a prime of form $2^a 3f - 1$ with positive integers a, f, let E be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, and let b be an positive integer. Suppose that the order of $E(\mathbb{F}_{p^2})$ is $(p+1)^2$. Our task is computing a random 3^b-isogeny from E. More precisely, we construct a function that takes E, b, and points in $E(\mathbb{F}_{p^2})$ as input and outputs the codomain of a 3^b-isogeny φ from E and the images of the points in input. In addition, we require that the isogeny φ is chosen uniformly at random from 3^b-isogenies from E whose kernel is cyclic.

For the case that $E[3^b]$ is not in $E(\mathbb{F}_{p^2})$, which is our case, two methods are known for computing such a function by $\mathbb{F}_{p^2}$-operations. First is taking a random order-3 point for each intermediate curve and computing a 3-isogeny with kernel generated by that point. Second is radical isogenies by [8]. We use the second in our implementation. The reason is as follows. Taking a random order-3 point needs scalar multiplication by $(p+1)/3$ in an elliptic curve. Using radial isogenies replaces this with a computation of a cube root in $\mathbb{F}_{p^2}$. As shown later, a cube root in $\mathbb{F}_{p^2}$ can be computed by an exponentiation by an integer approximately p in size. Therefore, using radical isogenies is more efficient than taking a random order-3 point.

A.1 Image under radical isogenies

It is well-known that an elliptic curve defined by $y^2 + a_1xy + a_3y = x^3$ has a point $(0, 0)$ of order 3. A radical isogeny between elliptic curves of such a form is the following formula.

Proposition 1 ([8, Section 4]). *Let E be an elliptic curve defined by $y^2 + a_1xy + a_3y = x^3$ and α be a cube root of $-a_3$. Then the codomain of an isogeny with kernel $\langle(0, 0)\rangle$ is isomorphic to $E' : y^2 + a_1'xy + a_3'y = x^3$, where $a_1' = -6\alpha + a_1$ and $3a_1\alpha^2 - a_1^2\alpha + 9a_3$.*

This formula is derived from the composition of the following two isogenies. First is an isogeny from E with kernel $\langle(0, 0)\rangle$ derived from Velu's formula. Second is an isomorphism that sends a point of order 3 to $(0, 0)$. The choice of α from the cube roots of $-a_3$ corresponds to the choice of a point of order 3 in the isomorphism. This choice determines the codomain of the next 3-isogeny.

We need to compute the image of a basis of $E_A[2^b]$ in our encryption function. For this, we use x-coordinate-only computation as in SIKE [2]. I.e., we compute the x-coordinates of P, Q, and $P + Q$ for a basis P, Q of the 2^b-torsion subgroup of the domain curve in each isogeny. This is more efficient than computing the full coordinates of P and Q because there are 4 values to compute in the full coordinates, while x-coordinate-only computation needs 3 values. We can obtain a formula of the image of a point under the isogeny in Proposition 1 by the construction described in the above paragraph. In particular, the first isogeny sends $(x, -)$ to $((x^3 + a_1a_3x + a_3^2)/x^2, -)$ and the second $(x, -)$ to $(x - a_1\alpha + 3\alpha^2, -)$.

A.2 Cube root of $-a_3$

We can taking α at uniformly random from the cube roots of $-a_3$ as follows.

The cube of $-a_3$ roots are in $\mathbb{F}_{p^2}$ since the 3-torsion subgroups of the elliptic curves which we use are defined over $\mathbb{F}_{p^2}$. Therefore, the multiplicative order of $-a_3$ divides $(p^2-1)/3$. Recall that we use p of form $2^a 3f - 1$ with a small cofactor f. If f is not divisible by 3, then $-a_3$ to the power of the inverse of 3 modulo $(p^2-1)/3$ is a cube root of $-a_3$. We can randomize this by multiplying a random cube root of unity.

In the case that f is of form $3^e f'$ with f' prime to 3, we can compute a cube root of $-a_3$ as follows.

1. Pre-compute a generator g of 3^e-torsion part of $\mathbb{F}_{p^2}^\times$, the inverse I_1 of $(p^2 - 1)/3^e$ modulo 3^e, and the inverse I_2 of 3 modulo $(p^2-1)/3^e$.
2. Let t be $(-a_3)^{((p^2-1)/3^e)I_1}$.
3. Compute the discrete logarithm h of t to the base g.
4. Then $(-a_3/t)^{I_2} g^{h/3}$ is a cube root of $-a_3$.

The computational cost of the discrete logarithm above is not large since f is small. However, that f is not divisible by 3 is preferable. The optimal choice of a and f depends on the computational costs of Richelot isogenies and the cube root. We leave this as future work.

A.3 Explicit algorithm

We use the same key compression as SIKE [2] (our proof-of-concept implementation uses the key compression function in the implementation of FESTA [3]), therefore the domain and the codomain of a 3^b-isogeny are represented by Montgomery forms.

In addition, we require ciphertext $(E_1, (P_1, Q_1); E_2, (P_2, Q_2))$ to satisfy that $[2^{a-1}]P_1 = (0,0)$ and $[2^{a-1}]P_2 = (0,0)$ because this property makes the computation of a glueing isogeny a little more efficient (see FromProdToJac in richelot_isogenies.py in the implementation of FESTA).

Transforming an elliptic curve of a Weierstrass form to a Montgomery curve is easy. In particular, given an elliptic curve E of a Weierstrass form and an order-4 point P on E, we can compute a Montgomery curve isomorphism to E in which the x-coordinate of the image of P is 1, so the image of $[2]P$ is $(0,0)$. Such a Montgomery curve uniquely exists [30]. We give an explicit algorithm in Algorithm 9.

In summary, our explicit procedure to compute a part of public key computed by a chain of 3-isogenies is as follows.

1. Decompress a public key and obtain a Montgomery curve E_A and a basis P_A, Q_A of $E_A[2^a]$.
2. Take a point R uniformly at random from the order-3 points on E_A.
3. Let ι be an isomorphism from E_A to an elliptic E'_A curve of form $y^2 + a_1xy + a_3y = x^3$ sending R to $(0,0)$. Compute E'_A and the x-coordinates of the images of P_A, Q_A, and $P_A + Q_A$ under ι (Algorithm 7).

4. Compute the codomain E_2' of a 3^b-isogeny ϕ_2 and the x-coordinates of the images of P_A, Q_A, and $P_A + Q_A$ under $\phi_2 \circ \iota$ (by using Algorithm 8 repeatedly).
5. Let κ be an isomorphism from E_2' to the Montgomery curve E_2 such that the x-coordinate of the image of $[4]P_A$ under $\kappa \circ \phi_2 \circ \iota$ is 1. Compute E_2 and the x-coordinates x_P, x_Q, and x_{P+Q} of the images of P_A, Q_A, and $P_A + Q_A$ under $\kappa \circ \phi_2 \circ \iota$ (Algorithm 9).
6. Compute points P_2 and Q_2 whose x-coordinates are x_P and x_Q, respectively.
7. If the x-coordinate of $P_2 + Q_2$ is not x_{P+Q} then change Q_2 with $-Q_2$.
8. Compress (E_2, P_2, Q_2) as a part of ciphertext.

Note that there are other forms of elliptic curves having a formula of radical 3-isogenies. A formula on Hessian curve is given by [6] and Montgomery curve by [30]. The most efficient choice depends not only on the efficiency of radical formulas but also on a formula of isogenies between abelian surfaces. We leave finding the best choice of these formulas as future work.

Algorithm 7 Weierstrass to curve for radical 3-isogenies

```
Require: An elliptic curve $E : y^2 + a_1xy + a_3y = x^3 + a_2x^3 + a_4x + a_6$, $(x_0, y_0) \in E$
    of order 3, and a set $X$ of the $x$-coordinates of points on $E$.
Ensure: $E' : y^2 + a_1'xy + a_3'y = x^3$ isomorphic to $E$ in which the image of $(x_0, y_0)$ is
    $(0, 0)$ and the set $X'$ of the $x$-coordinates of the image of points with $x$-coordinates
    in $X$.
 1: Let $f(x, y)$ be $y^2 + a_1xy + a_3y - x^3 + a_2x^3 + a_4x + a_6$.
 2: Let $g(x, y)$ be $f(x + x_0, y + y_0)$ .
 3: Let $c_1$ be the coefficient of $x$ in $g$ and $c_2$ be the coefficient of $g$.
 4: Let $h(x, y)$ be $g(x, y - c_1/c_2x)$.  // $h(x, y)$ is of form $y^2 + a_1'xya_3'y - x^3$.
 5: Let $E'$ be the elliptic curve defined by $h(x, y) = 0$.
 6: Let $X' = \emptyset$.
 7: for $x$ in $X$ do
 8:     Append $x - x_0$ to $X$
 9: end for
10: return $E'$ and $X'$.
```

Algorithm 8 Radical 3-isogeny

```
Require: An elliptic curve $E : y^2 + a_1xy + a_3y = x^3$ and a set $X$ of the $x$-coordinates
    of points on $E$
Ensure: The codomain $E' : y^2 + a'_1xy + a'_3y = x^3$ of an isogeny from $E$ with kernel
    $\langle(0, 0)\rangle$ and the set $X'$ of the $x$-coordinates of the image of points with $x$-coordinates
    in $X$.
1: Sample $\alpha$ uniformly at random from the cube roots of $-a_3$.
2: Let $a'_1$ be $-6\alpha + a_1$.
3: Let $a'_3$ be $3a_1\alpha^2 + a_1^2\alpha + 9a_3$.
4: Let $X' = \emptyset$.
5: for $x$ in $X$ do
6:     Append $(x^3 + a_1a_3x + a_3^2)/x^2 + a_1\alpha - 3\alpha^2$ to $X$.
7: end for
8: return $E' : y^2 + a'_1xy + a'_3y = x^3$ and $X'$.
```

Algorithm 9 Weierstrass to Montgomery

```
Require: An elliptic curve $E : y^2+a_1xy+a_3y = x^3+a_2x^3+a_4x+a_6$, the $x$-coordinate
    $x_4$ of an order-4 point on $E$, and a set $X$ of the $x$-coordinates of points on $E$.
Ensure: The Montgomery $E'$ curve isomorphic to $E$ in which the image of $(x_4, -)$ is
    $(1, -)$ and the set $X'$ of the $x$-coordinates of the image of points with $x$-coordinates
    in $X$.
1: Let $x_2$ be the $x$-coordinate of $[2](x_4, -)$.
2: Let $u$ be $1/(x_4 - x_2)$.
3: Let $A$ be $(a_2 + (a_1/2)^2 + 3x_2)u$.
4: Let $X' = \emptyset$.
5: for $x$ in $X$ do
6:     Append $x - x_2$ to $X'$.
7: end for
8: return $E' : y^2 = x^3 + Ax^2 + x$ and $X'$.
```